1990

Automated palletization of multiple box sizes.

Edric Chi-To. Lee
University of Windsor

Recommended Citation

Lee, Edric Chi-To., "Automated palletization of multiple box sizes." (1990). *Electronic Theses and Dissertations.* Paper 1250.

National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

AUTOMATED PALLETIZATION OF MULTIPLE BOX SIZES

by

Edric Chi-To Lee

A Thesis Submitted to the
Faculty of Graduate Studies and Research
through the Department of Industrial Engineering
in Partial Fulfillment of the requirements
for the degree of
MASTER OF APPLIED SCIENCE

At

the University of Windsor

Windsor, Ontario, Canada
1990

ISBN 0-315-60990-7

Lee, Edric Chi-To

Automated Palletization of Multiple Box Sizes

Due to absence of pagination, please omit appendices.

Please note also the addition of unnumbered pages into the prefatory material.

[illegible]

I hereby declare that I am the sole author of this thesis. I authorize the University of Windsor to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Edric Chi-To Lee

I further authorize the University of Windsor to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Edric Chi-To Lee

To my mother

ABSTRACT

The focus of this paper is on the physical aspect of the robotic palletization of multiple box sizes. Three algorithms were developed to cope with three different types of palletization problems. They are boxes of uniform height, boxes grouped by similar height, and boxes of different heights. A robot sequence program, with the aid of a vision system and a conveyor system, has been developed to implement the first algorithm and to carry out the loading process. The main objectives of this study are to reduce both the work in process area and the palletization time, and to maximize the space used on the pallet. Practical limitations are set and reflected in the algorithm. The results of this study are presented and compared with other results from existing algorithms.

ACKNOWLEDGMENTS

I wish to take this opportunity to give thanks to Dr. G. Abdou for his guidance and support throughout the research. I would also like to thank Dr. R. S. Lashkari and Dr. V. M. Huynh for reviewing my thesis and providing useful suggestions. Much thanks are also extended to Mr. Tom Williams for his technical help to the completion of the physical model. Special thanks to Ms. Jan Finlay for her help in the final preparation of the thesis.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

APPENDICES

CHAPTER I

INTRODUCTION

The palletization problem involves interlocking similar or non-similar boxes on a pallet. The main objective of the palletization problem is to maximize the use of pallet space with a combination of boxes. There are basically three methods used for palletizing. The first is manual pallet loading which is mainly applied to light loads, slow-speed operation and non-rigid environments.

The second is high-speed mechanized palletization for a non-flexible environment. This non-flexible environment is characterized by the use of a standard method for loading uniform size boxes on to pallets. This type of operation can be found in any canned food factory, such as Green Giant and Campbell, where all cartons of cans are of a standard size and there is only one standard way of loading these boxes on to pallets.

The third and most recent method is robot palletization which is best used in a medium-speed, non-rigid environment. This environment is characterized by the random arrival of non-similar boxes where there is no standard pattern of boxes on the pallet. UPI is an example of this slow or medium operation and non-rigid environment. In manual material handling, the operator is normally required to determine the box type and its allocation while in the robot palletizer method, the measuring functions are usually carried out by vision systems. This increase the efficiency of the whole system and makes it more automatic. Besides the methods of palletization, different dimensions, especially the height of boxes, and the space used for work-in-process, WIP, are also factors of the palletization problem. The term "work-in-process" in this study is defined as the temporary storage of boxes waiting to be loaded on to a pallet between operations. These factors may affect the layout of the manufacturing work area and the efficiency of the system.

In this research, the physical implementation of a three-dimensional palletization system with boxes of similar height will be studied. A vision system will be used to measure the two-dimensions, length and width, and the orientations of boxes on the conveyor. Then this data will be fed to the main program to determine where the boxes should be allocated. A transmission program will be used to give instruction to the robot to pick up the box and to load it on to the proper location. Moreover, the heuristic models dealing with similar height group boxes and boxes of different heights will be discussed. Also, a number of technological constraints on the robotic palletization process will be considered in the algorithms.

CHAPTER II

LITERATURE SURVEY

Because of its importance in manufacturing systems, the palletization problem has received considerable attention. However, only a few studies have been reported. These studies have developed in one basic direction. They all have attempted to apply various techniques for solving the mathematical aspect of the palletization problem. Only two papers by Tanchoco and Puls [6] and Tanchoco and Penington [7] have attempted to deal with the physical aspect of the problem. The two-dimensional palletization problem with boxes of uniform height was also the major concern of these studies.

A comparison between the existing systems and the proposed algorithms is illustrated in Table 1. A summary of each paper is presented as follows :

Tanchoco and Agee [1] presented a unit load handling system within a manufacturing system with operations such as packaging, palletizing, storage and shipping. Their presentation highlighted the many interactions between different alternatives in the system. These alternatives included box sizes, pallet sizes, palletizer load height, lift

Table 1. Features of previous works and proposal methods

FEATURES AUTHORS	LOADING METHOD	AVAILABILITY OF BOXES	WIP OR HOLDING AREA	SIZE OF BOX	BOX HEIGHT	METHOD USED	DIMENSION	TOL.	GRIPPER INTERFERENCE	NON-ORTHOGONAL LAYOUT OF BOX
Tanchoco & Agee (1981)	mechan. pal'er.	at all times	for all boxes	one	stndd. or fixed	———	2-D	no	———	yes
Smith & De Cani (1980)	manual	at all times	for all boxes	one	stndd.	exact & non- exact methods	2-D	no	———	yes
Kulick (1982)	manual	at all times	for all boxes	one	stndd.	simulation	2-D	no	———	yes
Hodgson (1982)	manual	at all times	for all boxes	un- limited	stndd.	dyn. prog. & heur.	2-D	no	———	yes
	manual	at all times	for all boxes	un- limited	diff.	inter. approach	3-D	no	———	yes
Hodgson, Hughes & Martin-Vega (1983)	manual	at all times	for all boxes	un- limited	stndd.	dyn. prog. & heur.	2-D	no	———	yes
Tanchoco & Puls (1986)	robot pal'er.	stoch- astic	for all boxes	maximum of five types	stndd.	three- block algorithm	2-D	yes	yes	yes
Tanchoco & Penington (1988)	robot pal'er.	stoch- astic	for all boxes	un- limited	stndd.	three- block algorithm	2-D	yes	yes	yes
Steudel (1979)	———	———	———	one	stndd.	heur.	2-D	——	———	yes
Christofides & Whitlock (1977)	———	———	———	one	stndd.	dyn. prog.	2-D	——	———	yes
Lee (1990)	robot pal'er.	stoch- astic	only as needed	un- limited	stndd.	heur.	3-D	yes	no	yes
	robot pal'er.	stoch- astic	only as needed	un- limited	some diff.	heur.	3-D	yes	no	yes
	robot pal'er.	stoch- astic	only as need	un- limited	all diff.	heur.	3-D	yes	no	yes

KEYS : mechan. - mechanized ; pal'er - palletizer ; stndd. - standard ; diff. - different
inter - interactive ; dyn. prog. - dynamic programming ; heur. - heuristic ; tol. - tolerance

truck capacity, warehouse storage, etc. The type and size of the pallet were specified and the best pallet loading pattern was selected. The limitation, though, was that all boxes were assumed to have a uniform size.

Smith and DeCani [2] described a rectangle-packing algorithm for a two-dimensional packing problem. A comparison between exact and non-exact packing methods was also included. The authors concluded that the non-exact program gave a guaranteed optimum layout and also required less computer time and storage than the exact program.

A simulation program which dealt with the interlocking pattern of the stacking shipper on a pallet was developed by Kulick [3]. The program only calculated an interlocking pattern of maximum space efficiency and did not guarantee the optimum pallet utilization. In the program, the height was restricted to the vertical dimension of the shipper on the pallet and the length was, of necessity, the longer of the two horizontally stocked sides.

Hodgson [4] approached the palletization problem using a combination of the principles of dynamic programming and heuristics. An interactive approach was also constructed by Hodgson for dealing with the three-dimensional palletization problem. The approach discussed two ways for loading pallets. The first was to pack boxes of common height. The second was to load the pallet with columns of boxes in order to make up stacks of boxes such that the stacks were no higher than the maximum allowable pallet height. Hodgson [5], in his next paper, presented two ideas for reducing the CPU time for running his dynamic programming based heuristic algorithm. The first idea was to develop a bounding function for limiting experimentation by solving relaxations of the palletization problem. The second idea was to take the largest box that had to be loaded and place it in one corner. Then the pallet load was built around the corner box.

Tanchoco and Puls [6], and Tanchoco and Penington [7] approached the pallet interlocking pattern problem in a different manner. They focused on the physical aspects of implementing a palletization system. An actual palletizing operation was described for a demonstration system developed using an IBM RS-1 robot. The robotic pallet loading system used the three-block palletization algorithm to generate a set of pallet loading instructions. The program calculated and stored pallet loading instructions for as many as five different box-pallet configurations. It was assumed that any one pallet could be loaded with only one box size and that multiple layers of boxes on a pallet could be positioned similar to the first layer. They then allowed different sized boxes to be loaded on a pallet. An algorithm was derived for determining the formation of boxes on each pallet level according to the corresponding coordinates for the placement of boxes. The palletization system derived was not limited in any way to the number of box types defined and used, the dimensions selected, or the sequence of box arrivals. The algorithm by Tanchoco and Penington [7] will be co pared to the two-dimensional pallet interlocking heuristics developed in the first phase of this project in terms of WIP and palletizing time.

The two-dimensional cutting stock problem is a two-dimensional equivalent of the palletization problem. The cutting stock problem deals with cutting specified lengths of material from given lengths of stock to minimize waste. Steudel [8] described a heuristic algorithm based on dynamic programming for solving two-dimensional cutting stock problems in which all the small rectangles were of the same dimensions and nonguillotine cuts were allowed. Christofides and Whitlock [9] used a tree-search algorithm for the solution of the two-dimensional constrained cutting problem. The algorithm limited the size of the tree-search by deriving and imposing necessary conditions for the cutting pattern to be optimal.

One of the major concerns when stacking boxes up on the pallet is stability. Low stability means boxes easily topple or slide off the pallet. Löschau [10] tried to solve this problem by developing different methods to investigate the stability criterion of column stacks. He used the critical inclination angle as the criterion for the rigid block (boxes loaded on the base of the pallet), and deviation from the center of gravity for the flexible bar model (boxes loaded on top of other boxes).

These publications have provided insight to the palletization problem. However, in order to develop a generalized algorithm which is more applicable to real life industry, further studies are required. There are three areas considered important to the problem which have not been properly discussed in these studies. The first area concerns the two-dimensional palletization problem, where an upper layer of a pallet cannot be loaded with boxes unless the lower layer is completely filled. Although Hodgson [4] briefly discussed the three-dimensional problem in his paper, no algorithm was provided and his work, at that time, was still under experiment. The second area concerns other performance measures, such as WIP, palletization time, and resources utilization affecting the palletization process. The third area concerns the random pattern of the incoming sequence of boxes.

CHAPTER III

PURPOSES OF THE STUDY

The main objective of this research is to develop the three algorithms for different combinations of box sizes. These algorithms are expected to provide more efficient solutions to the palletization problem. However, these solutions depend on both the dimensions of the boxes and the random sequence of the boxes coming into the system. These algorithms are also expected to achieve other goals such as reduction of WIP space and palletization time.

The three algorithms have been developed to deal with different complexity levels of the palletization problem: boxes of uniform height, boxes grouped by similar height, and boxes of different heights. A physical model has been developed for implementing the first algorithm. The algorithm interacts with a conveyor system, a vision system and a robot system. Different sequences have been used in running this physical implementation model for testing the effectiveness, efficiency and practicality of this algorithm in a real-world environment. The results from running this implementation have been compared with previous works. The comparison, though, was only used for the first algorithm. Because of the uniqueness of the second and third algorithm, and the fact that no algorithm on a three-dimensional palletization problem has ever been reported, the comparison was based on the existing data.

In summary, the main objectives of the research were:

1. developing a three-dimensional palletization algorithm for boxes of uniform height.
2. implementing the first algorithm into a physical simulator model with a vision system, a conveyor system, and a robot system.
3. comparing results obtained from running the physical implementation model and existing methods in the literature.
4. developing a three-dimensional palletization algorithm for boxes grouped by similar height
5. developing a three-dimensional palletization algorithm for boxes of different heights.

CHAPTER IV

PROCEDURES

The procedures for this study consisted of the following steps:

1. An algorithm for three-dimensional palletization problems with boxes of uniform height was developed.
2. A physical implementation model was developed by interacting the algorithm from step 1 with a vision system, a conveyor system and a robot system. The vision system was used for measuring purposes, the conveyor system for transporting boxes, and the robot system for moving boxes to the WIP area and to the pallet.
3. Different sequences were used to run the physical model. The resulting performance measures included palletization times, total WIP, maximum WIP and utilization percentages.
4. Comparisons were conducted between the results from step 3 and previous works in the literature.
5. From the results of step 1, a new algorithm was developed to model the palletization problem of boxes

grouped by similar height.

6. Another algorithm was developed to expand the results from step 5 to deal with the palletization problem of boxes of different heights.

CHAPTER V

PROPOSED ALGORITHMS

In order to make comparisons with previous works and to fulfill the objectives of this research, three algorithms were developed and will be presented in this chapter.

Each of these algorithms contains different levels of complexity. The first one deals only with the three-dimensional palletization problems with boxes of similar height. These boxes may be different in length and/or width but their heights must be uniform. The second algorithm goes one step further by considering boxes grouped by similar height. That is, some boxes have a similar length and width but are grouped according to their different heights and others are different in length and width but have the same height. This is a little more complicated than the first algorithm because boxes coming into the system may have different heights. Thus the second algorithm must be able to cope with this situation and to decide the arrangements of box-layers on to the pallet.

Since the second algorithm is not the most complicated case in palletization as it deals with groups of boxes with the similar height, a third algorithm has been developed for the case which deals with boxes of totally different dimensions. In this case, the box types are different in all three dimensions. This is the most complex case among all palletization problems.

The size of the pallet and the number of different box types for these three algorithms are not important factors as each of the algorithm is capable of supporting any size of pallet and as many box types as the user requires. The height of all boxes must be specified as the vertical dimension of the box and cannot be mistaken for the length or the width. This is especially important to the first and second algorithms when all or some of the boxes have uniform heights. Another restriction for these algorithms is that the total area of the boxes in each layer should not exceed the pallet area or the area of the layer itself. In other words, no overhanging is allowed. Also, the stacks of boxes must be no higher than the maximum allowable pallet height. The remaining of the chapter will discuss the heuristic models of the three algorithms.

Heuristic model refers to simple (common sense) rules for obtaining a good (near-optimal) solution to large, complex optimization problems [11]. In this case, they describe the step by step procedures for achieving the objectives.

5.1 Three-dimensional problem with boxes of similar height

The first algorithm deals with the three dimensional palletization problem with boxes of a similar height. This algorithm is also employed in the physical implementation. Its goal is to achieve the following objectives:

1. maximize the space use on the pallet
2. minimize the space use in the work in process
3. minimize the palletization time by the robot
4. minimize the total material handling time which combines the total palletization time and the time taken for loading boxes to the WIP area

All of these objectives very much depend on the random introduction of different types of boxes into the system. In the first objective, different sequences may result in different maximum utilization percentages and some of them may not be able to obtain the optimal solution to the problem. Similarly, some sequences require moving more boxes to the WIP area. This increases both the space used in the WIP area and the number of robot moves, in turn, increases the palletization time.

Different constraints were set to define limitations or boundaries under which the objective functions can be obtained. There are three kinds of constraints incorporated in the algorithm. They are presented as follow.

5.1.1 Pallet constraints

The first five sets of constraints of this model relate to the dimension of the pallet. These constraints are:

1. the total area of all the boxes on each layer k must not exceed the area of the corresponding layer of the pallet.

2. the total lengths and/or widths of boxes lying along the length of layer k of the pallet must not be larger than the length of the corresponding layer.
3. the total lengths and/or widths of boxes lying along the width of layer k of the pallet must not be longer than the width of the corresponding layer.
4. the area of the upper layer must has an equal or smaller area than the lower layer. To be more specific, the length and width of the upper layer must be no longer than the length and width of the lower layer, respectively.

The orientation of a box on the pallet is not necessarily length to length or width to width as some boxes may have their widths lying along the length and perpendicular to the width of the pallet.

5.1.2 Sub-area constraints

The total area available on each layer of the pallet is partitioned into small sub-areas, and boxes are placed on one of these sub-areas. The following constraints restrict the loading of boxes with larger dimensions, for both length and width, than the space of the sub-area.

1. For normal orientation, the length of the box lies parallel to the length of the pallet, the box must have an equal or

smaller length than the length of the sub-area.

2. For normal orientation, the box must have an equal or smaller width than the width of the sub-area.
3. For the box that has been turned 90° from its normal orientation, it must have an equal or smaller length than the width of the sub-area.
4. For the box that has been turned 90° from its normal orientation, it must have an equal or smaller width than the length of the sub-area.

5.1.3 Work in process constraint

The work in process constraint limits the total area of boxes in the WIP area, at any time during the operations, to be less than or equal to one maximum surface area of a pallet. A complete step by step formation of the algorithm is recorded on a flow chart as presented in Figure 1.

Since the boxes arrive in random sequence, a comparison between boxes is required for determining which box should go on to the pallet first and which should go later. The first comparison takes place between boxes on the conveyor and in the WIP area. If there is no box in the WIP area, the comparison is between the first two boxes on the conveyor. Under normal circumstances, the box with the larger area is chosen to load on to a sub-area of the pallet. However, an

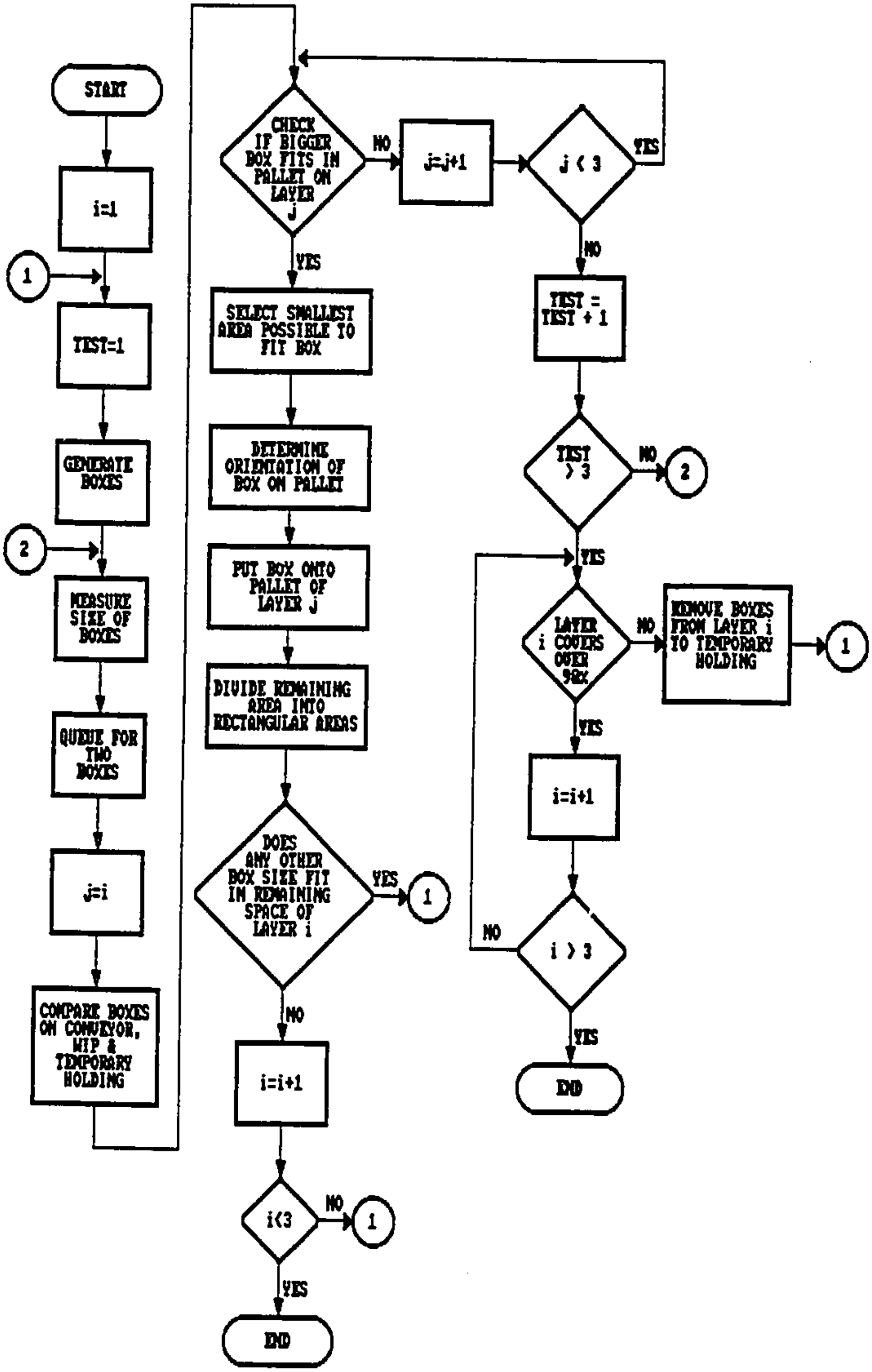


Figure 1. Flow chart of the first algorithm

abnormal circumstance occurs when the smaller box happens to be the same dimension as one of the sub-areas on the pallet. In that case, the smaller box will be selected over the larger box. If a tie exists, the box in the WIP area will be chosen. Otherwise, the first box to come into the system will be selected.

When loading a box on to the pallet, some conditions should be checked. If all the remaining sub-areas of layer i on the pallet are too small for even the smallest-sized box as indicated by the data variables, the loading starts on layer i+1. If layer i+1 is greater than three, which is the maximum number of layers on the pallet depending on the requirement of the system, the loading is completed for that pallet. In other words, the pallet is filled. Availability of areas in the upper layer is directly determined by the surface area covered in the lower layer. That is, if a box with the dimensions of 24 inches by 20 inches is loaded onto layer 1, layer 2 automatically provides space. Therefore, the upper layer is always equal to or smaller than the surface area of the lower layer.

In a case where a larger box is chosen in the early part of the algorithm but is too large for any of the remaining sub-areas on layer i, the algorithm checks layer i+1 and/or i+2 given that either one is less than three. If there are enough sub-areas available to fit the box, the robot will load the box on to one of these sub-areas of the pallet. Otherwise, the algorithm will either check the WIP and test a smaller box, if there is one, in those sub-areas or it will compare the smaller box with another box on the conveyor. Again, the box with the larger area is selected for loading on to the pallet. The same checking procedures are repeated for this box. If there is no sub-area available, the algorithm goes through the same checking procedures with the third box. If the same result occurs, it checks whether or not the layer has been loaded with more than 90 % of the boxes. If not, the robot removes some of the boxes already on the pallet to the WIP area to make room for the largest boxes in either the WIP area or the pick up area. The WIP area, at any time during the operation, is limited to be less than or equal to a maximum surface area of a pallet. If the total area of the boxes in the WIP area, at one point in time, is larger than the size of a pallet, those boxes will be placed on to a second pallet.

If there is more than one sub-area on the pallet available for the chosen box, the algorithm selects the smallest one to fit the box. The box is loaded on to the bottom left hand corner of its corresponding sub-area. The orientation of a box on the pallet is also an important factor in maximizing the coverage of pallet space. The orientation is determined by comparing the two sets of calculations, as follows;

$$|\lfloor lp_z/L_i \rfloor| * |\lfloor wp_z/W_i \rfloor|$$

and

$$|\lfloor lp_z/W_i \rfloor| * |\lfloor wp_z/L_i \rfloor|$$

The two variables, lp_z and wp_z, represent the length and the width of the sub-area z on the pallet and L_i and W_i represent the length and the width of box i to be loaded on to the pallet, respectively. The set providing the larger value represents the correct orientation of box i on the pallet since the larger value means that more boxes of the same size can be loaded on to the sub-area with that orientation. For instance, if the first set, $||lp_z/L_i||*||wp_z/W_i||$, gives the larger value, the orientation of the box is such that its length (L_i) lies parallel to the length of the sub-area (lp_z). On the other hand, if the second set results in the larger value, the orientation will be the other way around whereby the length of the box lies along the width of the sub-area.

After a box is loaded on to a sub-area of the pallet, the remaining empty area is partitioned into new rectangular sub-areas. The rule of thumb is to partition it in such a way that the new sub-areas are able to fit as large a box as possible. These procedures will keep repeating until no more sub-area is available on the pallet.

5.2 Three-dimensional problem with boxes grouped by similar height

In this algorithm, boxes are palletized on to the pallet in layers as shown in Figure 2. Extra objective functions and constraints are added, as follows.

1. Maximize critical inclination angle, θ_k.
2. Minimize deviation from the center of gravity, d.
3. Maximize the stack height.

The first two objective functions are developed for

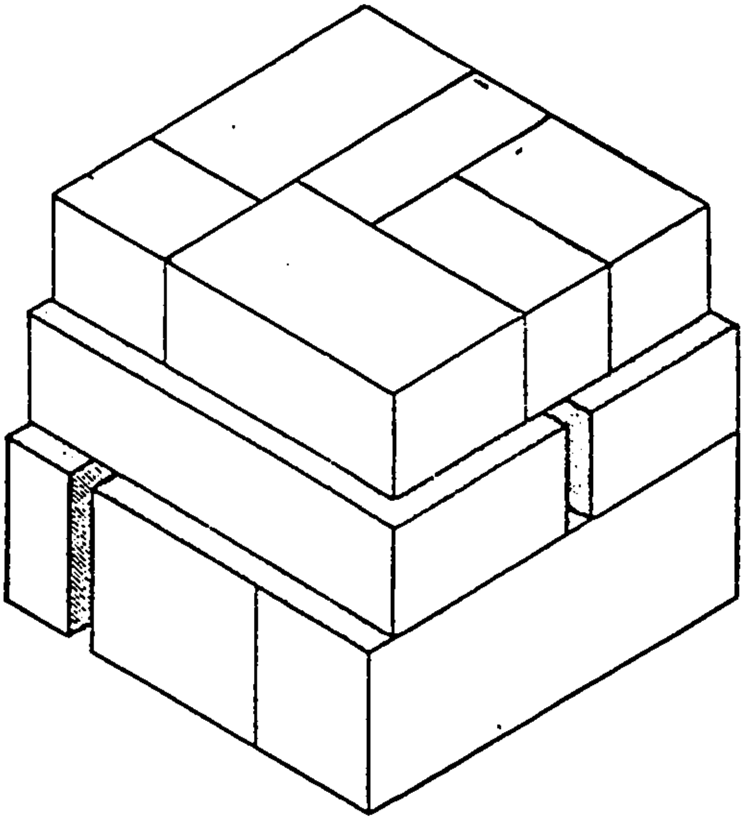


Figure 2. Layered pallet load

selecting a better stability criterion between boxes. The larger value of θ_k means that the box is more difficult to slide off the pallet. Likewise, the smaller deviation from the center of gravity refers to greater stability against toppling. The third added objective function is for maximizing the total height of the stack, which is the total height of boxes on each layer. The added constraints are reflected in the calculation of the stability criteria and are discussed in the next section.

In this algorithm, once a box height is loaded on to a layer of the pallet, this layer must maintain this box height for the rest of the process. Boxes of other heights are not permitted to palletize on to the same layer. This may lead to a very high number of boxes in the WIP area. Therefore, the algorithm has no limitation on the total WIP area. Since the boxes do not have a uniform height, the layers will have different heights and the total number of layers on the pallet will depend on the total stack height which cannot exceed the maximum pallet height.

One of the differences between the first and the second heuristic models is in the selection process of boxes to be loaded on to the pallet. When dealing with boxes of different heights, the stability of stacks becomes one of the major factors. Löschau [10] presented different methods for obtaining the stability criteria and they are discussed in the following sections. Löschau considered the first layer of the pallet as a rigid block since it was located on the base of the pallet, and the other layers as a flexible bar model because these boxes were loaded on top of other boxes which vibrated during transportation.

5.2.1 Stability Criterion For A Rigid Block

The stability criterion developed by Löschau for a rigid block is presented in the following equation;

$$\frac{W}{\bar{H}} \geq \tan \theta_k \quad \text{where } \bar{H} = H - h$$

Since the width of the box is the smallest dimension, the variable, W, is considered in the above formula. H and $\bar{H}$ are the maximum height of the pallet and the height of the space above the stack of a box height, h, respectively. In order for the box not to slide off the pallet, $W/\bar{H}$ must be larger than the tangent of the critical inclination angle. $L/\bar{H}$ is not considered because it is redundant. Since L is always larger than W, if $W/\bar{H}$ satisfies the stability criterion, so is $L/\bar{H}$. However, the reverse does not always be true.

Since the critical inclination angle, θ_k, denotes the angle of a box that causes it to slide off the pallet by vibration during transportation, tan θ_k, therefore, refers to the friction coefficient of the box. The larger angle or friction coefficient means the box is more stable on the pallet. As a result, the following rules are developed for choosing the first box to load on to the first layer of the pallet. The choice is based on which box provides the larger inclination angle. The selection process starts when there are already two boxes in the system. The rules are:

1. if both boxes have equal height, the one with the greater width is chosen. This is because the greater width results in a larger inclination angle.
2. if the length and the width of the two boxes are similar, the one with the greater height is chosen. (i.e. greater h, smaller $\bar{H}$, greater tan θ_k).
3. if the two boxes have a similar length but are different in both width and height, the

comparison is between $W_1/\overline{H}_1$ and $W_2/\overline{H}_2$. The greater of the two results in the larger θ_k.

4. if the width of the boxes is similar, but the length and height are different, the comparison is the same as rule number 3.
5. the last rule is when all three dimensions are different. Once again, rule number 3 applies for the selection process.

Once a box is selected to load on to the first layer of the pallet, only boxes with the same height can be loaded on to the same layer. In other words, if the incoming boxes have the same height as those on the first layer, they will be loaded on to the first layer according to rule number 1. Otherwise, the different heights of the boxes are compared and selected for the second, third or other layers by applying the flexible bar model.

5.2.2 Stability criterion for a flexible bar model

As discussed in reference [10], this method is for deriving the stability criterion of a box on any layer of the pallet other than the first. In this method, more factors are involved. The values of the critical inclination angle, θ_k, is different for each layer. They can be obtained by applying the following equation;

$$\theta_k = 41*\frac{W*H'}{\overline{H}'} = 41*\frac{W * H'}{(H' - \sum_{i=2}^{K} h_i)}$$

Since the flexible bar model is used for loading boxes on to the pallet starting on the second layer, H' and $\overline{H}'$

become the height of the pallet minus the height of the box on the first layer, and the space above the stack of boxes, respectively. If a box is loaded on to the second layer, K equals 2 and $\bar{H}'$ equals $H'-h_2$ with h_2 as the height of the box on the second layer. For the third layer, $\bar{H}'$ equals $H'-(h_2+h_3)$ with h_2 and h_3 as the heights of the boxes on the second and the third layers, respectively. No new layer can be added on to the pallet if $H' < \sum_{i=2}^{K} h_i$ because the total height of the layers must not be greater than the maximum height. Other factors involved are the elastic modulus, E, and the polar moment of inertia, I, of a box. E is constant if the material contained in each box is the same and this value can be obtained from the elastic modulus table. The formula for calculating I is presented as follows;

$$I = \frac{W^3 * h}{12}$$

Once these values are calculated, the horizontal deviation d_c and the deviation from the center of gravity, d, can be calculated by the following equations;

$$d_c = \frac{1 - \frac{V}{15} + \frac{V^2}{720} - \frac{V^3}{71280}}{1 - \frac{V}{6} + \frac{V^2}{180} - \frac{V^3}{12960}} * \frac{\bar{H}' * \sin\theta_k}{2}$$

$$d = d_c * \frac{\bar{H}'}{H'}$$

The value V is given by;

$$V = \left(\frac{j}{E * I}\right) * \bar{H}'^3 * \cos\theta_k$$

where j is a constant which represents the weight force per unit height. The stability of the boxes was given by Löschau as follows;

$$d \le \frac{W}{2}$$

As the deviation from the center of gravity decreases, so does the possibility of a box toppling off the stack. Therefore, when comparing two boxes, the box with the smaller deviation from the center of gravity is selected. For example, if the material of the product is steel, the elastic modulus, E, of the steel equals $210*10^9$ N/m^2, and the weight force per unit height, j, equals 2940 N/m. Also, the maximum height of pallet minus the height of the box on the first layer, H′, equals 100 cm. The dimensions of the two boxes are as follows:

L_i=25cm ; W_i=15cm ; h_i=8cm ; L_{ii}=15cm ; W_{ii}=8cm ; h_{ii}=20cm

The moments of inertia of the boxes are:

$$I_i = \frac{(0.15)^3*(0.08)}{12} = 2.25*10^{-5}\ m^4 \quad ; \quad I_{ii} = 8.533*10^{-6}\ m^4$$

The critical inclination angles are:

$$\theta_{ki} = 41 * \frac{0.15*1.00}{(1.00-0.08)^2} = 7.27^o \quad ; \quad \theta_{kii} = 41 * \frac{0.08*1.00}{(1.00-0.2)^2} = 5.13^o$$

The V values are:

$$V_i = 4.806 * 10^{-4}$$

$$V_{ii} = 8.370 * 10^{-4}$$

The values for the horizontal deviation of the two boxes are:

$$d_{ci} = 0.0582\ m$$

$$d_{cii} = 0.0308\ m$$

Finally, the deviation from the center of gravity can be calculated as follows:

$$d_i = 0.0535\ m \quad ; \quad d_{ii} = 0.0286\ m$$

Both deviations are less than half of their corresponding widths, so that both boxes fulfill the stability criterion and since the second box has a smaller deviation from the center of gravity, it is selected to load on to the second layer of the pallet.

The overall procedures of the algorithm are similar to the first algorithm. A brief explanation of the steps follows:

1. check whether or not there are already boxes on the pallet. If not, compare the first two boxes with the rigid block method to select one box for the first layer. If yes, check if the incoming box has the same height as the box on the pallet. If yes, go to step 2. Otherwise, go to step 3.
2. check if there is a sub-area on pallet available for the box. If there is, load the box on to the proper sub-area. If not, place the box in the WIP area. Then measure another box and go back to step 1.
3. check if the maximum height has been reached. If yes, continue accepting boxes and repeat step 1. If not, use the flexible bar model to select boxes to load on to the upper layers.

These three steps replace the procedures for comparing the area of boxes in the first algorithm. The remaining steps, except for the two discussed below, are more or less the same for both algorithms.

In the second algorithm, there is no restriction on how many trials are tested for each iteration (three in the first algorithm). This, in turn, eliminates the steps that follow the third test which is unsuccessful in the first algorithm. Moreover, the restriction of a maximum pallet size in the WIP area is eliminated in this algorithm because the number of boxes required can be very high if several of the incoming boxes do not have the same height as any of the layers on the pallet. Nor can they be loaded onto a new layer.

5.3 Three-dimensional problem with boxes of different heights

For the third algorithm, all types of boxes have different heights and they will be stacked on to the pallet as shown in Figure 3.

There are many different ways of interlocking these boxes on the pallet and four of them are discussed in this section. The algorithm will follow the description of one of these methods. The first method is to load the boxes on to the pallet as long as there is a sub-area available. In this method, the interlocking is totally random and consists of no particular pattern. Since the dimensions of all types of boxes are known, the second method will take advantage of this fact and wait for a combination of boxes that gives the same total height before beginning to load boxes on to the pallet. Eventually, this method will interlock, as well as layer, the boxes. The shortcoming of this method is that the WIP is

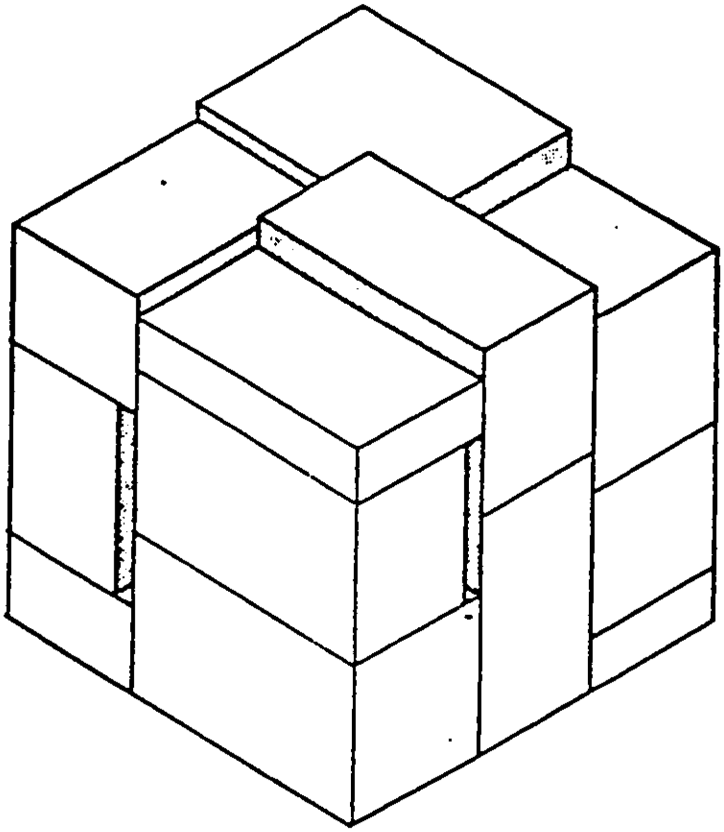


Figure 3. Stacked pallet load

expected to be very high. The third method is columnar stacking in which all boxes are stacked in columns with no interlocking. In this method, it is best to load boxes with as large an area as possible on the bottom because boxes on the top must have a smaller or equal dimension. If the stacking begins with small boxes on the bottom, the choices decrease for selecting boxes to load on top of them. The fourth method is a combination of the first two methods whereby a comparison is made between two boxes, and the box with the greater stability is selected for loading on to the pallet as long as there is a large enough sub-area. If the total height of two boxes, one on the pallet (box A) and the other on the conveyor or in the WIP area (box B), equals the height of another box (box C) placed next to box A, box B will have first priority to be loaded on top of box A to produce a partial layer with box C on the pallet, provided box B fulfills the corresponding flexible bar model stability criterion. This method implements the third algorithm because it saves the WIP area compared to the second method and produces partial layers which the first method may not be able to achieve.

For implementation of this method, the three steps discussed in the second heuristic model is replaced by the following steps.

1. Check if one of the two boxes produces a partial layer with a box on the pallet. If yes, this box is selected if it also fulfills the flexible bar model stability criterion. Otherwise, go to step 2.
2. Check which layer has enough sub-area to fit either one of the two boxes. If the sub-areas are too small, go to step 3. If the sub-areas are adequate, compare the stability criteria of the two boxes by applying the proper method.
3. If there is no one particular sub-area on any layer large enough to fit either one of the two boxes (in other words, the sub-area can only fit one of them), no comparison can be made between

the boxes. The algorithm will put these boxes in the WIP area and let other boxes come into the system. Then the process reverts to step 2.

CHAPTER VI

PHYSICAL IMPLEMENTATION

The purpose of a physical implementation is to test the practicality of the proposed algorithm when it actually runs in a physical environment. It also serves as a simulation model. A simulation follows the operation of a real-world model, pattern, process, or system over time. It involves the generation of results from a physical system and inferences are drawn concerning its operating characteristics. A physical implementation requires the interaction of different systems to carry out the operation process. It also takes the form of a set of assumptions which can be expressed in logical forms between the entities of the system.

In this research, the physical implementation involves the application of a vision system, a robot system, and a conveyor system to load different box sizes on to a pallet. A BASIC program has been developed on an IBM PC-XT compatible computer to control these systems. Three different box sizes are considered in this implementation and the boxes are assumed to have guillotine cuts, that is, right angle cuts. Also, due to the possibility of various forces encountered on the corners, non-orthogonal layouts of the boxes on the pallet are not considered. The original size of the pallet is 48 inches by 40 inches but is scaled down in the program to 12 cm by 10 cm. The pallet can be stacked up to three layers. The sizes of the three box types are 40 inches by 24 inches (10 cm by 6 cm), 24 inches by 20 inches (6 cm by 5 cm), and 20 inches by 12 inches (5 cm by 3 cm). The boxes are assumed to have a standard height of 12 inches (3 cm) and they are scaled according to the size and the lift capacity of the robot gripper. Except for the lengths of the type 1 boxes, no other dimension is greater than the maximum opening of the robot gripper, and each box weighs less than the lift capacity of the robot. All boxes are painted black to increase the contrast between the boxes and the background.

The systems used in this implementation play significant roles in the performance of the model. Each of these systems contains a variety of equipment which operates to make the palletization model more practical. In the following sections, specifications of the equipment and the operation procedures are described.

6.1 The vision system

The vision system contains three different components: a light source, a camera and a 4 bit-image digitizer. They are described as follows.

6.1.1 light source

In order for the vision system to operate successfully, the scene must have enough light cast on it and be free from shadowing effects. This light source also must be bright enough to activate the photo cells used in the conveyor model. However, since the photo cells are sensitive, the light source cannot be too bright. Otherwise, its brightness may cause the photo cells never to stay on. For this particular implementation, no light source, other than the room light, is needed since the light in the room provides adequate lighting.

6.1.2 Camera

An HV-62 HITACHI vidicon camera is used since it is the only camera available in the department. A detailed specification of this camera is presented in Table 2. It operates by allowing the reflected light from an object to pass through a lens to form an image on its image plane. A electrical charge is created when a photon strikes the sensor on the image plane. Thus, an analog image can be obtained directly from the camera by projecting an image on to the image plate. The camera takes a 2-dimensional image by raster scan and converts the lightwave into voltage peaks. A digitizer, which will be described in detail in the next section, converts these signals into numbers. It then gives each voltage peak a numerical value. When a 16-gray level digitizer is utilized, a dark spot will result in zero value since there is no generated voltage. A moderately bright surface gives a range from 7 to

10 while an extremely bright surface achieves a value of 15. Therefore, the contrast between the box and the background plays a very important part in the accuracy of the measurement of box dimensions.

Table 2. Vidicon camera specifications

ITEM	SPECIFICATIONS
Vidicon tube	2/3 inch electrostatic focusing, 20PE20 or equivalent
Deflection system :	
Vertical frequency	Line lock, 60 Hz (U,C) or 50Hz (E,K)
Horizontal frequency	15.75 KHz (U,C) or 15.625 KHz (E,K)
Interlace	Random interlace
Resolution (Horizontal)	Over 500 lines at center
Video output signal	Composite 1 Vp-p (Sync 0.3 Vp-p)
Sensitivity	10 to 100,000 lux
Ambient temperature	-10 to +50 degree Celsius
Power consumption	7 W
Dimensions	65(H) x 99(W) x 200(D) mm
Weight	Approximately 1.3 kg

Some disadvantages of this type of camera are that it distorts the image if the voltage is not stable and it is very sensitive to temperature. Despite these shortcomings, the camera is still used in the system not only because it is the only camera available, but because the working environment can be maintained to enable the camera to perform at its best.

6.1.3 4 bit-image digitizer

An in-house developed image digitizer board is installed in the computer and used in this implementation. The digitizer resolves 256 x 256 pixels and provides up to 16 levels of gray scale of precision. It can be considered the heart of the vision system. Its function is to convert the analog video signal to the digital signal which can be processed by the computer. It operates by first accepting analog video signals from any video source. The next step is to convert these signals to digital data. The conversion task begins by partitioning the image into pixels. Then the digitizer assigns an integer gray level value to each pixel corresponding to the measurement of its brightness. Once the conversion is completed, it stores this data in the digitizer memory as a two-dimensional array of gray levels. Then the image processing can start.

A special feature of this digitizer is that it uses digitizer memory instead of computer memory to store data. This allows the vision system to store a digitized image at the full video rate and later make individual pixels available to the computer at memory access speeds. This technique results in a much faster reading of the image (approximately 60 scans/sec) than the one using computer memory [12].

6.1.4 Vision system operation

When the vision system is being called upon to measure, the image of the box is taken by the HV-62 Vidicon camera and digitized by the digitizer using 16 levels of gray scale. The output, which is already in digital form, represents both the image of the box and its background, and is stored in the memory whose address is known. Once the image is captured on the screen, a BASIC program performs the analysis in the memory.

The image is stored in a buffer at &HA000 address in such a way that each byte contains two pixels. Thus, 32767 bytes are available to store 256 x 256 pixels. Each pixel on the screen corresponds to 4-bit data in the frame buffer. For simplification and faster scanning, the buffer is assumed containing an image of 128 x 128 pixels.

The next step is to isolate the image of the box from its background. There are different techniques which can be used to accomplish this task. A thresholding method is selected for scanning, for the following reasons:

1. The image is quickly processed
2. The method is easy to implement in the hardware
3. The method is widely used in industrial application [13].

With this technique, each pixel is given a value and is compared to the threshold value of 8. A threshold can be considered as a reference or standard level in the 16 levels of gray scale to which all measurements of the pixels are compared. A pixel having an intensity below a threshold is regarded as an edge, or part of the box, and a pixel with an intensity above the threshold is considered as part of the background. This method analyzes the picture by scanning from the top left corner of the screen across to the right, and down to the bottom until the first black pixel is found.

In order to ensure that the first black pixel is an edge, the program continues to scan the next four pixels. If all of them are black, that means the first black pixel found is indeed an edge instead of just a noise pixel. In this implementation, the location of a box is identified around the center part of the screen.

In order to speed up the image processing and to save considerable time, instead of beginning the scan from the top, the scan begins somewhere in the middle of the screen. Once the first black pixel is found, the program records the position of this pixel for later use. Then the program continues scanning across the screen to the right until it reaches a white pixel. To ensure that it is a white pixel, the next four pixels are scanned. The position of this pixel is also recorded by the program. The difference between these two position values gives the width of the box in pixels. An illustration of this operation is shown in Figure 4.

The next step is to find the length of the box. This procedure is illustrated in Figure 5. Since the positions of the box widths are already known, the center of the box can also be determined. This center point is set as the reference

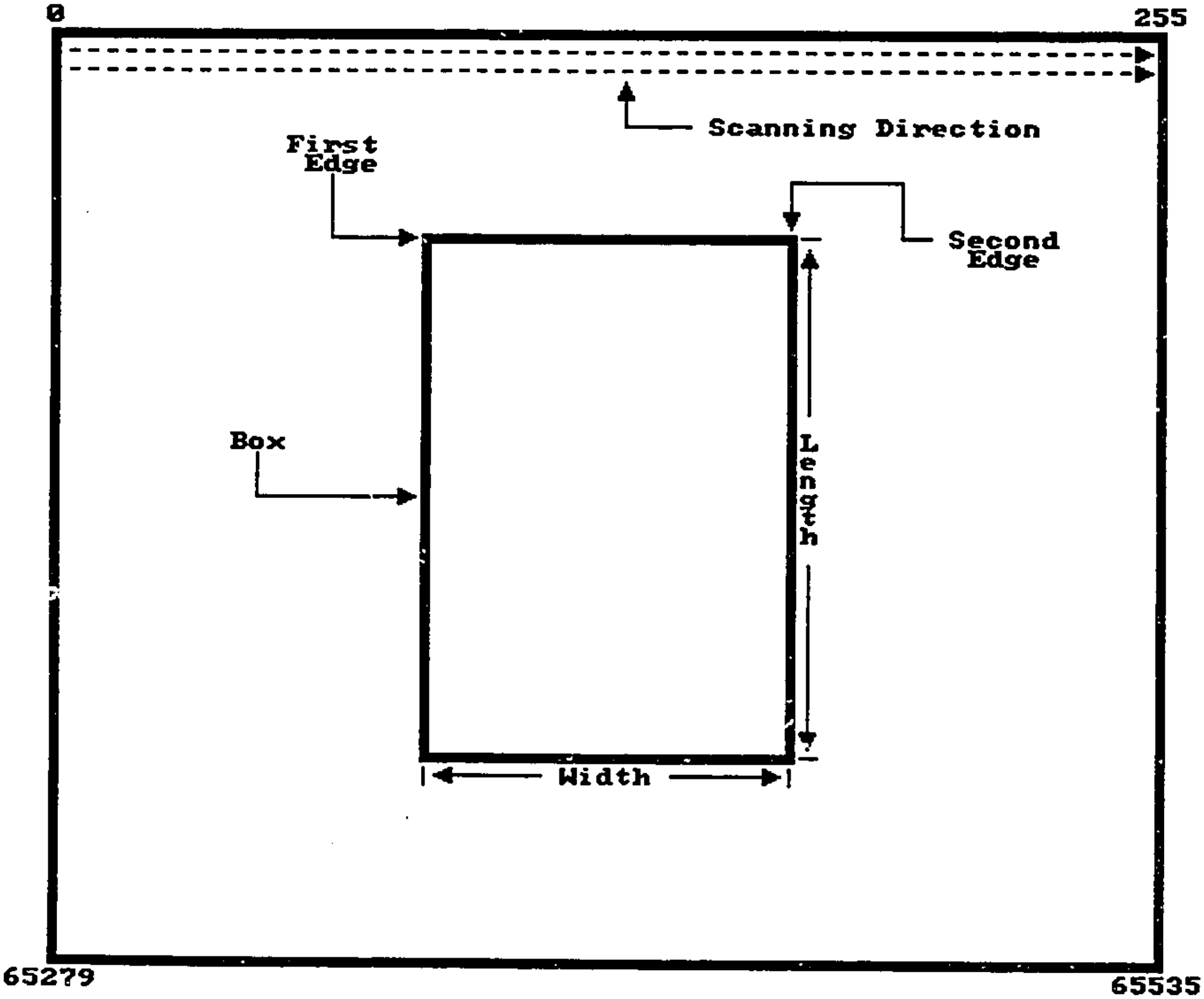


Figure 4. Method of scanning width of box

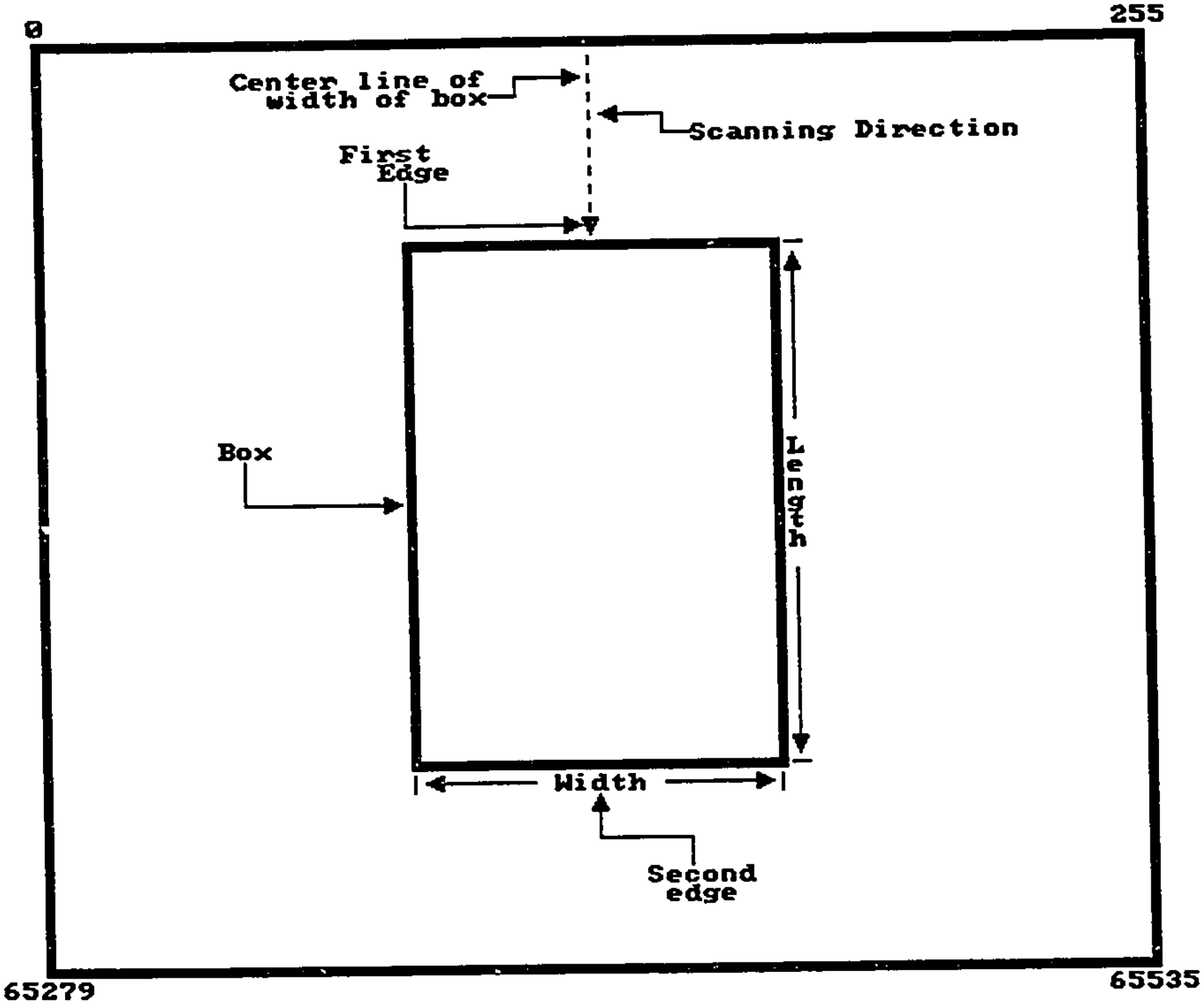


Figure 5. Method of scanning length of box

column, or X, pixel. The scanning process then begins from the pixel to determine the edges on the two widths of the box. These scanning procedures are basically the same as those which have just been described. The only distinction is that scanning is from the top of the screen and continues to the bottom along this column. The differences between the position values in this case will not directly give the length of the box. This is because the two position values provide the exact location of the pixels on the screen, but not their Y, or row, values. Since there are 128 pixels on each row, in order to find out the Y values of these two edges, the user must divide each of these position values by 128 and take only the integer part of it. For example, a position value of 10024 becomes 78 in the Y value. Once the two Y values are calculated, the length of the box in pixels can be determined by taking the difference between these two Y values. Note that it is not necessary to calculate the X values of the two edges of the box because they lie along the same row on the screen. If these values are desirable, they will be calculated by dividing each position value by 128. Then the residual of each result is multiplied by 128. For instance, the X value of 10024 equals $\{(10024/128) - ||10024/128|| \} * 128 = 40$.

The accuracy of the measurement by the vision system in both vertical and horizontal directions is +/- 2 pixels for every box. Thus, the larger box is measured more accurately because the ratio between the error and the dimension of the box is smaller. At this stage, then, the program has obtained the dimensions of the box. However, these dimensions are in pixels and their values are much larger than the exact dimensions of the boxes. In order to scale these values down, a scaling factor is used. Once this scaling process is completed, the program is able to determine which box type the measured box belongs to.

This program offers a trade off between flexibility and speed. If speed is required, the location of the box should be fixed as in this implementation. That is, its location must be fixed or in a certain area range on the screen. On the other hand, if flexibility is desired, the speed of the image processing is expected to be slow because of the greater range of the scanned area.

6.2 The robot system

In order to simulate industrial activities, a Scorbot-ER V robot is used for material handling. Even though the robot was built for educational purposes, it is flexible enough to perform many industrial tasks. The components of the system and their operations will be discussed in the following sections.

6.2.1 Robot arm and controller

The robot arm has five degrees of freedom plus a servo gripper. The controller has the capacity to control up to eleven motors including the six motors for the robot arm and the gripper. These eleven motors are divided into two groups, A and B, plus an independent motor C. The robot controller is also a multi-tasking system which can run up to twenty independent programs simultaneously. And with the serial RS 232C port as the communication channel, the controller is suitable for interacting with any IBM PC/XT/AT/PS-2 or compatible computer. Detailed specifications and basic dimensions of the robot arm and the controller are presented in Table 3 and Table 4.

6.2.2 Teaching pendant

Scorbot-ER V can be controlled either by a teaching pendant or through programming languages. The teaching pendant allows the user to operate the robot without any computer interface. It does not allow the use of all the built-in functions that the controller is capable of; it is best used for locating positions. A list of the specifications of the teaching pendant can be seen in Table 5.

6.2.3 Software control

When a micro-computer is interfaced to the controller, the robot can be programmed to perform operations by special languages of its own - ACL (Advanced Terminal Control Language) and ATS (Advanced Terminal Software). The robot movement is activated, only, by the vendor software which needs a transmission program to interact with any external file. This transmission program opens the communication port

Table 3. Robot arm specifications

ITEM	SPECIFICATIONS
Mechanical Structure	Vertical articulated robot Five degrees of freedom plus servo gripper
Operation range :	
Base rotation (Axis 1)	325°
Shoulder rotation (Axis 2)	+130° - 35°
Elbow rotation (Axis 3)	+/- 150°
Wrist pitch (Axis 4)	+/- 130°
Wrist roll (Axis 5)	Unlimited
Radius of operation	610 mm (24.4 inches)
Gripper opening :	
Excluding rubber pad	74 mm (3 inches)
Including rubber pad	65 mm (2.5 inches)
Sensors	Gripper can measure object's size
Hard Home	On all axes
Feedback	Optical encoders on all axes
Drive system	Electrical DC servo motors
Motor capacity (axes 1-5) :	
Peak torque (stall)	15 oz in
Power for peak torque	70 W
Maximum payload (excluding the gripper)	1000 g
Repeatability	+/- 0.5 mm
Maximum path velocity	600 mm/sec.
Robot weight	Approx. 10.8 Kg
Flexible hose	Factory installed for pneumatic end effectors applications

Table 4. Controller specifications

ITEM	SPECIFICATIONS
Controller type	Real time multi-tasking controller. Stand alone controller.
No. of servo axes	Maximum - 11 axes Standard - 8 axes
Groups of control	11 axes can be divided into 2 groups (A and B) plus independent axes (C)
Axes control	Pulse Width Modulation (PWM), 20 KHz
Path control	Point To Point (PTP) Continuous Path (CP) -linear, circular, mathematical curves, (10 ms. response time)
Path control profiles	Paraboloid, trapezoid, open loop free running
Interpolation function	Articulation interpolation, linear interpolation, circular interpolation
Motion speed definition	Either by speed or by time travel
Control parameters	Integral, proportional, differential, offset, open or closed loop
CPU	Motorola 68010
EPROM	256 K bytes
Work RAM	64 K bytes
User RAM	32 K bytes - standard 128 K bytes - expanded (optional)
No. of program lines and positions	1. Standard RAM-4000 prog. lines or 1500 positions (or any combination)

Table 4 continued

	2. Expanded RAM-12000 prog. lines or 6000 positions (or any combination)
No. of programs in user RAM	Hundreds
Multi-tasking	Run up to 20 independent programs simultaneously. User can edit other programs at the same time.
I/O	16 Inputs 16 Outputs : 4 Relay outputs (N.O. or N.C.) 12 Open collector outputs
Home	Limited switch for each axis
Communications	RS - 232C serial port
Emergency Brake	Red switch for emergency brake
Programming Methods	1. Advanced Control Language (ACL) : Using any terminal or an IBM PC/XT/AT/PS-2 with the Advanced Terminal Software (ATS) More than 100 commands available 2. SCORBASE : Using an IBM PC/XT/AT/PS-2
Positions teaching method	1. Using ACL or SCORBASE (without a TP - Teaching Pendant) 2. Using a TP
Coordinate systems	1. Articulated (joints) coord. system 2. XYZ coord. system
Sensors	Sensors interrupt capabilities
Protection	Hardware : 1. Current limit on each axis

Table 4 continued

	2. Automatic fuse on each axis Software : 1. Impact protection 2. Thermal protection 3. Boundary protection
Troubleshooting	1. Built-in test routine 2. LED indication on each servo axis and on each input/output 3. HDT: Hexadecimal debugger tool
Power requirements	110 VAC / 230 VAC, 60 Hz / 50 Hz, 500 Watts
Power switching	Power ON switch, green LED indication. Motors ON switch, green LED indication.
User's power supply	+12 VDC, 2 Amps
Motors' power supply	+24 VDC, 18 Amps
Weight	10 kg. (421 lbs.)
Dimensions	490 mm (L) x 455 mm (W) x 150 mm (H)

Table 5. Teaching pendant specifications

ITEM	SPECIFICATIONS
Display	LCD, 2 lines, 32 characters
Keypad	30 Multi-functional keys Color coded. Beep sound on each touch
Coordinate system	XYZ and robot joints
No. of controlled axes	11
Straight line movements	X+, X-; Y+, Y-; Z+, Z-;
Pitch control	1. End effector moves up or down while arm does not move 2. End effector stays fixed on its point and only changes its orientation while the other arm axes move
Speed setting	User can choose any value from 1 to MAXIMUM
Record positions	For each control group separately
Go position movements	To previously recorded positions
Run programs	Run preprogrammed user programs
Abort programs	Abort user programs
Control ON/OFF	Enabling/disabling of axes' control Emergency robot halt
Full status display	Coordinate system, selected group, messages

to transfer commands from a terminal to the robot controller. In this simulation model, an ACL program is written to manage all material handling operations with a transmission program. It is written in BASIC and sends information about the orientations and locations of boxes from the external computer to the controller. However, this transmission program has its shortcomings. The commands sent from the external computer must be transmitted one at a time because the controller cannot handle more than one command. That is, if the CPU (Central Processing Unit) of the computer runs faster than the transmitting process of the controller, a loop must be added between each process to delay the execution and wait for the former to finish before transmitting the next command. Otherwise, either the last command may be lost during the previous transmitting process and result in incorrect pick up and/or target locations, or an input/output device error will appear.

6.2.4 Robot system operation

In performing the material handling task, the robot follows a typical point-to-point (PTP) operation. It moves to a certain point until the box is within the reach of the gripper. The robot picks up the box and then moves either to the pallet or the WIP. Consequently, the robot will move back to its home position awaiting another task. For every movement of the robot, it must pass through a so-called dummy point in order to avoid any collision.

The program is developed in such a way that the task of changing the target point in the future can be implemented just by inserting the new coordinates X, Y, Z, P (Pitch), and R (Roll), or by increasing the coordinates' values from previous ones to form a new target point. During that conversion of coordinates to encoder position values, the program will calculate how many steps the motors have to run through in order to move the robot arm from one position to another. These calculations will result in running the motors simultaneously. One thing the user must keep in mind is that after the controller is turned on, the user must enter one of the two robot software languages and set the robot back to the hard home position. Otherwise, the position of the robot at the time the controller is activated will become the home position and all other coordinates will be altered.

6.3 The conveyor system

The purpose of this conveyor system is to transport boxes to the measuring and pick up areas. This conveyor model was built in-house using Fischertechnick components. It consists of two belts, two photo cells, and an interface board.

6.3.1 Conveyor belts

A conveyor belt, in general, is suited to material handling situations involving high volumes, unit loads, packaged products, fixed points, and repetitive cycles. It is considered as a power conveyor which normally used in heavy handling applications such as machine shops. Conveyor belt can operate on inclines of 30° or more, depending on the pallet size and shape and the type of belt surface. An advantage of this type of conveyor is its ability to handle almost any shape that will not overhang or roll off en route. The two conveyor belts used in this implementation are each 26 inches long and parallel. The conveyor belts are built parallel to the floor surface, that is, no inclination angle. Their speed and direction are controlled by the interface board which will be described later. Since the belts are made of Fischer components, they cannot support a heavy box.

6.3.2 Photo cells

Two photo cells are located between the belts for signaling the conveyor to stop for other operations. Once the conveyor is stopped, it either calls up the vision system to measure the dimensions of the box, or to signal the robot sub-system to unload another box from the conveyor to the pallet or the WIP area. The photo cells are controlled by the light source where bright light opens the cells and dim light closes them.

6.3.3 Interface board

The motor is used to run the conveyor belts, and along with the two photo cells, it is controlled by an in-house developed interface board. This interface board can control up to 16 motors and 16 switches. It allows the user to control the conveyor either clockwise or counter-clockwise with different speeds, and to control the opening and closing of the photo cells. Specifications of the interface are listed in Table 6.

Table 6. Interface board specifications and its applications

Data Acquisition and Control

for IBM PC/XT/AT and Compatible Microcomputers

32 Channels : 16 Analog Ouput and 16 Digital Input Interface Board

Features

* Plugs directly into IBM PC/XT/AT Bus
* 16 switching sensing Digital Input lines
* 16 -8 bit Analog output channels
* +6, -6 Volt Output Range
* Variable Speed Control
* Bidirectional Output
* 300 mA current source and sink capability
* Remote mounted off switch, sets all control channel output voltages to zero for safety to allow for program initialization

Applications

* Servo Control
* Laboratory automation
* Physical Simulators
* Activates alarms
* Annunciator lights
* Useful with A/D's

6.3.4 Conveyor system operation

Both the speed and direction of the conveyor can be controlled by a BASIC command "OUT 784-MOT1,127+S", where MOT1 and S are variables assigned to the motor number, and the speed and turning direction, respectively. The variable, MOT1, can be any integer number between 1 and 16 while variable S ranges from -127 to 127 with the two extremes representing the maximum speed in both directions and the middle point, 0, indicating stop.

The photo cells, on the other hand, are activated by a light source. A value of 0 shows that there is enough light cast on the photo cells while a value of 1 means either the working environment is too dark or something is blocking the light source from a photo cell. Therefore, as a box is moving past, its shadow will cause the photo cell to change from 0 to 1 and signal the conveyor to stop moving. However, the photo cells from Fischertechnick are too sensitive. That in regular light source, a box is still unable to trigger the photo cell and to change its bit value from 0 to 1. Therefore, two resistors, which can be adjusted from 0 to 50 ohms are added to reduce the sensitivity of the photo cells.

6.4 System integration

The integration of the three systems described earlier has been achieved by the appropriate interfaces to a microcomputer and the BASIC program (THESIS.BAS). Except for the program that performs the material handling operations, all other systems including the robot transmission program are subroutines of the main program. The main program itself is used for comparing boxes, determining the allocations and the orientations of the boxes on the pallet or on the WIP, and checking whether or not there is enough empty space available for another box to load on to the pallet. All of these subroutines and the main program are written in BASIC while the robot control program is written in the ACL language. The computer code of this program and the subroutines will be discussed in the next chapter. A detailed diagram describing the interfacing among the three systems is presented in Figure 6.

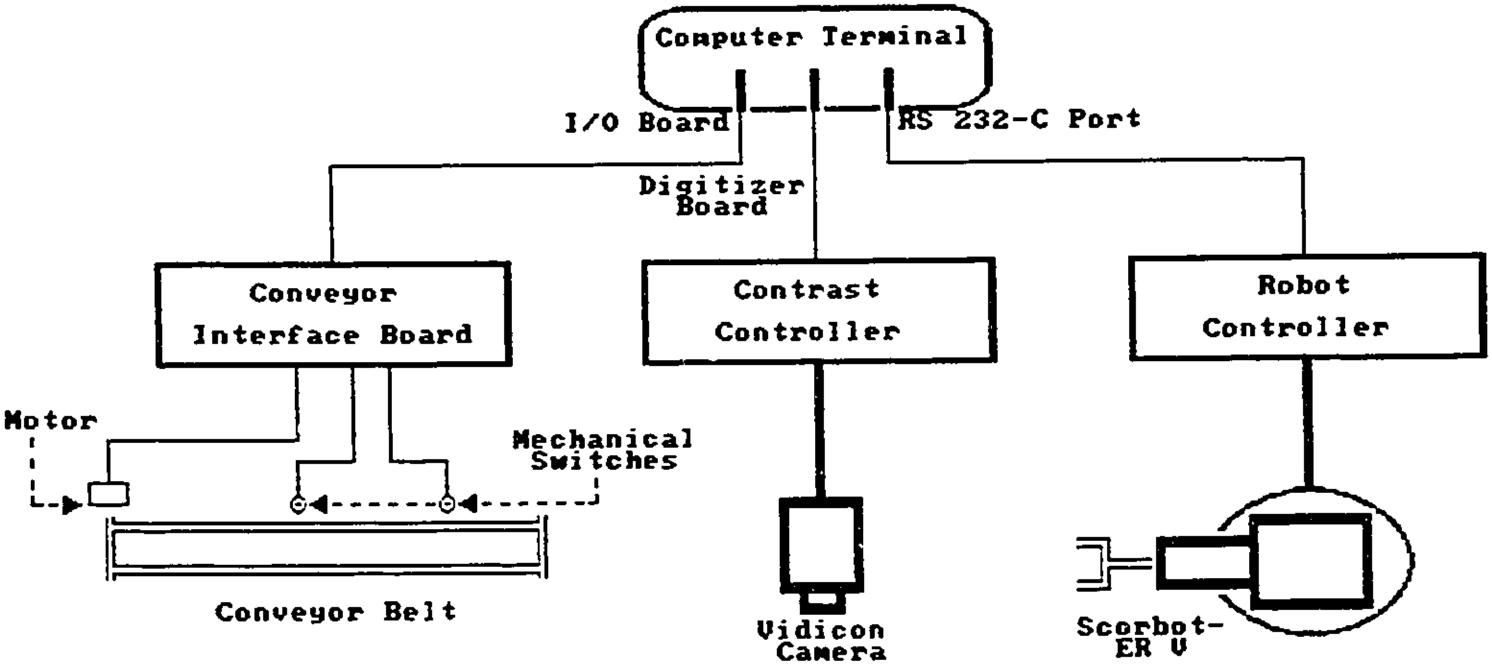


Figure 6. Equipment interfaces to microcomputer

The main program calls the conveyor subroutine to transport boxes along the conveyor until one of the photo cells is triggered. If the photo cell activated is located underneath the camera, the conveyor subroutine calls the vision subroutine in which the image of the box is captured, analyzed, and identified as either one of the known box types or as an erroneous box. If it is the other photo cell, which is located at the pick up area, that has been triggered, the conveyor subroutine will call the transmission subroutine to transmit the pick up and target coordinates to the robot controller, and activate the robot to move the box to the destination. Otherwise, the conveyor subroutine will send the signal back to the main program for other comparisons. This is because any box transported to the pick up area is not necessarily removed right away. In this case, when the box is finally required to be loaded onto the pallet or to the WIP, the main program will call the transmission subroutine directly with information about the pick up and target locations, and the orientation of the box. The transmission subroutine then transfers these data to the robot controller and runs the material handling program. Once the transmission process is completed, the subroutine calls the vision system to measure another box if there is one underneath the camera. Otherwise, the conveyor subroutine will be called to move a box to the measuring area.

The interaction between the three systems is designed in such a way that it allows the main program to control both the conveyor and the robot systems. The conveyor system, on the other hand, interacts with both the vision system and the robot system.

6.5 Overall system operation

The following section deals with the operation procedures. First, the program allows the user to enter the required data. This data, as shown in Table 7, contains values of the variable arrays that define the length and width of the pallet and the multiple box sizes used. These values of pallet's and boxes' dimensions can be scaled down.

Table 7. Input data

```
Length of pallet : 12
 Width of pallet : 10
No. of box types ? 3
Length of type  1 box ?
? 10
Width of type   1 box ?
? 6
Length of type  2 box ?
? 6
Width of type   2 box ?
? 5
Length of type  3 box ?
? 5
Width of type   3 box ?
? 3
```

For example, instead of typing in the values as 40 and 24 for the type 1 box, the values are stored as 10 and 6. It is assumed that all boxes are arriving continuously on the conveyor. When a box arrives under the camera, the vision system is activated and/or when a box moves to the pick up area, the robot arm is activated. The camera is located directly above the center of the conveyor. The beginning and the end of the conveyor are defined as the loading and the pick up areas, respectively. The pick up area, the pallet and the WIP area are all reasonably located within the maximum reach of the robot arm. A complete work area layout can be seen in Figure 7.

When all the data are entered, the next step involves receiving and recognizing an incoming box. The box is first transported from the loading area to the measuring location where a camera will capture its image and send the analog signal to the digitizer board for scanning and measuring the length and the width of the box. The measurements were not too accurate. Thus, the contrast between the box and the conveyor has to be increased. This has been achieved by placing a white sheet of paper underneath the box when it is first loaded on to the conveyor. A tolerance is also set to circumvent any errors introduced in measuring the dimensions of the box. This tolerance helps to determine which of the three types of boxes is presented in the measuring area. It determines a range of length and width for each of the three boxes in the data array. The tolerance in the implementation is set to +/- 0.4 cm which is approximately equal to the repeatability of the vision system in measuring the smallest box, type 3. It is used to set the tolerance because it has the relative error. As a result, the length of the type 1 box stored in the data array will be recognized as between 9.6 cm and 10.4 cm, and the width

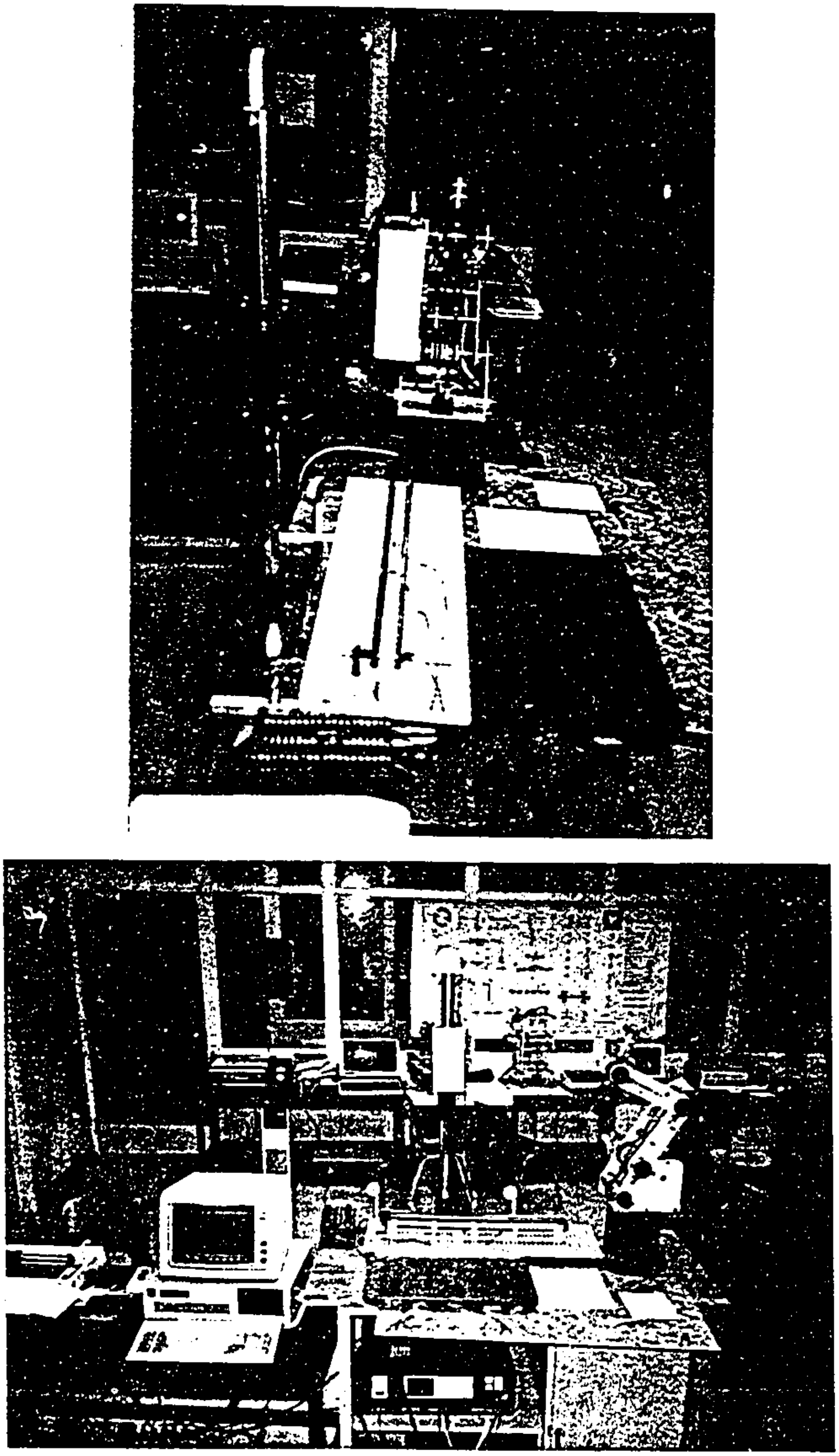


Figure 7. Work area layout

will be recognized as between 5.6 cm and 6.4 cm. Thus, a box measuring 10.13 cm by 5.81 cm will be recognized as a type 1 box, but a box 9.55 cm by 6.04 cm will not be recognized. If the dimensions of a box, taking the set tolerance into consideration, correspond to one of the three known box types, the box will be recognized and handled correctly. If the box dimensions do not match any of these three types, the program will request the user to remove this erroneous box from the conveyor. When setting the tolerance, the user must consider carefully all the dimensions of the boxes so that different types are not recognized as the same and an erroneous box is not recognized as one of the valid boxes.

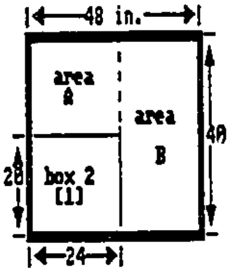
After the box is measured, it will be compared to all boxes in the WIP area and also to the box at the pick up area. If there are no boxes in either area, the conveyor transports the measured box to the pick up area. The orientation of the boxes on the conveyor is such that the box length is oriented perpendicular to the side of the conveyor belts. It is also assumed that the box dimensions, resulting from the vision system, are width and length of that box. Since the coordinate of the center of the measured of box can be determined, the program can send the robot accurately to the pick up area. Another assumption is that the distance between any two boxes on the conveyor is the same. Therefore, when the first box is stopped in the pick up area, the box behind it can be under the camera waiting to be measured. Once the measuring process on the second box is completed, the program compares its area with that of the first box. The box with the larger area is selected to load on to the pallet. If the first box happens to be smaller than the second one, the robot will move the first box to the WIP area. A set of arrays has been incorporated to store the types and locations of all boxes in the WIP so that the program knows where to send the robot to pick up the box, when it is needed.

Before any palletizing is carried out, the program is required to determine both the orientation of the box on the pallet and how the robot should grasp the box, either by the length or by the width. Once a box is loaded on to the pallet, three subroutines in the program will partition the remaining area of the pallet into sub-areas, as described in Chapter VII. The results are stored in another set of arrays for later use.

In order to describe this physical implementation of the palletization algorithm, a sequence of operations for completely filling one layer of a pallet will be described in detail. A complete schematic diagram showing this sequence of operations is presented in Figure 8. In this example, the boxes used to fill up the pallet are two type 1 boxes, six type 2 boxes, and four type 3 boxes. Assume the pattern of the random sequence of the boxes arriving into the system is 3-2-2-2-3-2-3-1-2-1-3-2-2-1-2-3-3-2-... After an initializing process and the input of data required by the user, the first box is transported along the conveyor until it interrupts the

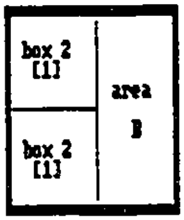
STEP 1

3-2-2-2-3-2-3-1-2-1-3-2
compare : 3 & 2
3 to WIP
2 to 1st layer of pallet



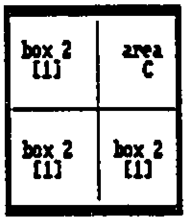
STEP 2

2-2-3-2-3-1-2-1-3-2
WIP : 3
compare : 3(WIP) & 2
2 to area A on 1st layer of pallet



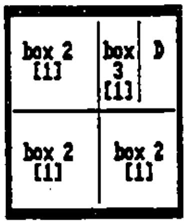
STEP 3

2-3-2-3-1-2-1-3-2
WIP : 3
compare : 3(WIP) & 2
2 to area B on 1st layer of pallet



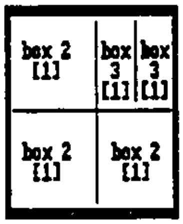
STEP 4

3-2-3-1-2-1-3-2
WIP : 3
compare : 3(WIP) & 3
3(WIP) to area C on 1st layer of pallet



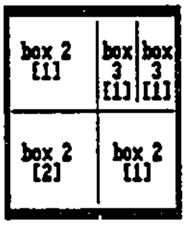
STEP 5

3-2-3-1-2-1-3-2
compare : 3 & 2
3 to area D on 1st layer of pallet



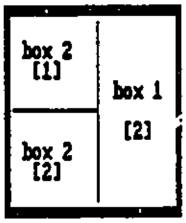
STEP 6

2-3-1-2-1-3-2
compare : 2 & 3
2 to 2nd layer of pallet



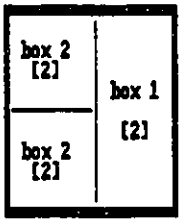
STEP 7

3-1-2-1-3-2
compare : 3 & 1
3 to WIP
1 to 2nd layer of pallet



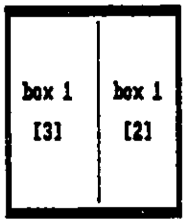
STEP 8

2-1-3-2
WIP : 3
compare : 3(WIP) & 2
2 to 2nd layer of pallet



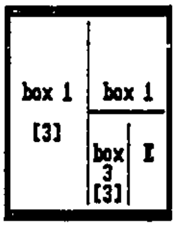
STEP 9

1-3-2
WIP : 3
compare : 3(WIP) & 1
1 to 3rd layer of pallet



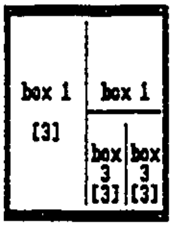
STEP 10

3-2
WIP : 3
compare : 3(WIP) & 3
3(WIP) to 3rd layer of pallet



STEP 11

3-2
compare : 3 & 2
3 to WIP
3 to area E on 3rd layer of pallet



STEP 12

WIP : 2
no comparison
2 to 3rd layer of pallet

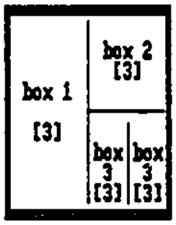


Figure 8 : Step by step illustration of example
(KEY : [?] - LAYER NUMBER)

first photo cell and activates the vision subroutine to perform its functions. The results show it is a type 3 box and the measurements is shown on the screen once the measuring process is completed. Next, the vision subroutine returns these results back to the main program for comparison. Since it is the first box, no comparison will take place. The main program will call the conveyor subroutine again to transport this box to the pick up area and, move the second box underneath the camera. The vision subroutine will once again be called to perform the analysis on the box. This time, the results indicate it is a type 2 box. The program will then compare this box with the one on the pick up area of the conveyor, and the second box is selected to load on to the pallet because it has the larger surface area between the two. Then, the main program will call the robot subroutine to remove the first box from the conveyor to the WIP and allows the second box to move to the pick up area. While the second box is being transported to the pick up area, a new box is moving to the measuring area.

Once this transportation is completed, the robot subroutine will transmit the pick up and the target coordinates of the box to the controller and run the ACL program. The system has now become a multi-tasking system by performing two different operations at the same time. The first task is to unload the type 2 box from the pick up area and palletize it on to the pallet. The second task is to run the vision subroutine to analyze the third box in the sequence moved to the measuring area. When the analyzing operation is finished, the program will go to another subroutine for partitioning the remaining area of the pallet into sub-areas as described in the heuristic model. Then, the third box, which is another type 2 box, is compared with the box in the WIP area. Once again, the type 2 box is selected for placement on the pallet because of its larger surface area. Since there are two sub-areas on the pallet, the program will first determine which sub-area the box should load on to. This will be the smaller sub-area because it has the same dimensions as the type 2 box. The procedures for palletizing this type 2 box will include signaling the conveyor subroutine to transport the box to the pick up area, calling up the robot subroutine to transmit the required information to the controller and to run the ACL program to palletize the box. This same process of selecting sub-area and loading procedures will be repeated as the fourth box in the sequence, also a type 2 box, arrives into the system.

At this stage, the first layer of the pallet has been covered by three type 2 boxes. The fifth box in the sequence arrives is of as type 3 box. Since it is the same type of box as the one in the WIP, the box in the WIP area is selected for loading on to the pallet before the one on the conveyor. While the robot is loading the box on to the pallet, the conveyor will transport the type 3 box to the pick up area and allow a sixth box to move into the measuring area. Then the vision subroutine analyze the sixth box as of type 2 box.

After this loading operation, the first layer of the pallet now has only one sub-area left which can fit only a type 3 box. This time, the newly measured box is identified as type 2 which is larger than the type 3 box in the pick up area. But since the first layer only has enough empty space to fit one type 3 box, the type 3 box lying on the pick up area is chosen over the larger type 2 box based on one of the rules in the algorithm. By completing this loading task, the first layer is filled. The second and third layers of the pallet can also be filled by repeating the same operations. However, with different types of boxes in the random sequence, the procedures would be different from the one discussed. These operations will continue until all three layers of the pallet are "filled up".

The term "filled up" does not always refer to a 100 % utilization. If the largest empty sub-area(s) left on the pallet is so small that even the smallest type of box cannot fit, or in another case, when there is only an area large enough to fit in the smallest box type, the pallet is considered "filled up". However, if the next few boxes in the sequence are all larger than the sub-area, the program will test only three of them rather than test all of the boxes and place them onto the WIP area. If none of the three boxes fit into the sub-area, the program quits and considers the pallet filled in order to fulfill the objectives of minimizing both the use of the WIP area and palletization time. However, the program will only quit if the pallet is already loaded with more than 90 % of the boxes. If it is not, another set of procedures will be used to take care of this problem. A detailed discussion of this procedure can be found in the heuristic model of the first algorithm. In this example, percent coverage is able to achieve 100 % utilization. At the end of a completed palletization, the pallet utilization and the total palletization time will be shown on the monitor screen, as in Figure 9.

```
BOXES ON PALLET ARE AS FOLLOWS :
 1 : 2
 2 : 2
 3 : 2
 4 : 3
 5 : 3
 6 : 2
 7 : 1
 8 : 2
 9 : 1
 10 : 3
 11 : 2
 12 : 3
MAXIMUM WORK IN PROCESS ARE AS FOLLOWS :
 1 : 3
THE PALLET IS FULL AND THE UTILIZATION IS 100 PERCENT
THE TOTAL TIME IS 00:06:37
```

Figure 9. Final performance measures

6.6 Implementation problems encountered

A few problems were encountered during the operation of this robotic palletization model for loading multi-size boxes on to a pallet. This section will describe the nature of these problems and explain the possible solutions. Some of these solutions were applied and resulted in a positive improvement to the system.

The first problem is the limited computer memory in the BASIC interpreter. The program used in this physical implementation requires a large three-dimensional array and many other two-dimensional arrays for storing data. As a result, the interpreter is unable to execute the program completely. A solution to this problem is found by compiling the THESIS.BAS BASIC program into an executable THESIS.EXE file. Therefore, the compiled version of the program provides enough memory to complete one cycle of execution which the BASIC interpreter failed to do. It executes at a much faster speed than the BASIC file in the interpreter. One shortcoming of the compiled file is that it takes up a large amount of memory on the diskette.

The second problem deals with the sensitivity of the photo cells. As mentioned before, the photo cells are so sensitive that resistors are required in order to decrease their sensitivity. However, when resistors are connected to the photo cells, the feed back signals are inconsistent, for example, signals suddenly change from 0 to 1 and stop the movement of the conveyor even when there is no box blocking the light source. This, many problems are generated and the palletization process fails. Mechanical micro-switches can replace the photo cells for signaling the conveyor to stop. Experience with currently working model confirms that the micro-switches are more reliable than the photo cells.

The third problem concerns with the inaccuracies of the vision system in measuring the length and the width of the boxes. This problem is solved by assigning a set tolerance for the two dimensions as explained in an earlier section. The user can change the tolerance limits within the program. However, these limits should be carefully selected, since they define the acceptance range of boxes. This tolerance limits may be determined based on working experience with a particular vision system. The fourth problem is the gripper interference. To minimize this problem, all boxes have already been defined as having guillotine cuts and a standard height. Even so, there are still possibilities of collision between the gripper and a box already placed on the pallet during the robot's operations. For example, when the robot is loading another box on to the same layer, the gripper might interfere with any box already placed on the pallet. This problem is solved by extending the robot gripper with two steel plates, 1 mm thickness, which the robot picks up the box. And when the gripper releases the box on to the pallet, it will open very slowly as the robot arm move up and away from the box. Thus, by the time the robot gripper is totally away, it leaves approximately a 2 mm gap between the two boxes on the pallet. Another solution may be applied to the problem if the substances contained in the box are not fragile. In this case, the robot may release and drop the box from a certain height above the pallet. The latter solution, though, will not be used in this implementation.

The last problem is the orientation of the box on the conveyor when it is ready for pick up. Due to the vibration of the conveyor and/or the improper loading of boxes on to the conveyor, the orientation of boxes could be shifted along one or two axes on the conveyor and/or placed on the conveyor at an angle. This could create a serious problem for the gripper when picking up the box. The three error types and their possible solutions are described below.

6.6.1 Transposition in X and/or Y axis

The first type of error is where the box shifts either to the X or the Y direction. The solution is to assign each type of box a reference point which is the center point of the box as the vision system captures and analyzes its image. This box is assumed to be situated perfectly in the assigned location. If there is no loading or vibrating error acting upon the box, the center points for the boxes should not be changed. In other words, changes in these points indicate a shifting of the axis in the boxes' orientations. If the row, or Y value, of a center point is different from its respective reference point, this indicates the box has shifted along the width of the conveyor. The program will calculate how much the box has been moved to the right or left and will adjust the gripper to the proper place for picking up the box. This, of course, assumes that there is no other change in the orientation of the box when it is being transported from the measuring area to the pick up area. A similar adjustment is needed when the column, or X value, of a center point does not match its respective reference points. However, in this implementation model, the box is stopped by a switch in the pick up area. Therefore, there is no transposition along the X axis.

6.6.2 Rotation about X and/or Y axis

When vibration and/or other factors cause the box orientation to rotate with an angle about the X and/or Y axis, the program is required to calculate this angle to correct the gripper coordinates. The procedure of calculating this angle will be fully explained in the next chapter. Once the angle is determined, the corresponding value is added or subtracted from the roll cartesian angle of the gripper. This correction allows the gripper to rotate the proper angle for grasping the box. If a box on the conveyor is rotated on an angle and is transposed on the Y axis, the robot has to adjust its Y value and rotation angle in order to cope with these errors.

Unfortunately, the existing vision system is not capable of measuring these angles. As presented in Table 8, the results are not consistent. Moreover, the smaller the angle, the bigger the error. This error is due to the digitizer accuracy in measuring the rotation angle. For instance, there is not much difference in interpreting an angle between 75° and 90°, 63° and 75°, etc. Therefore, instead of applying this solution to the program and risking outrageous

Table 8. Summary of measurement of different angles by the vision system

Actual Angle (In Degree)	Measured Angle (In Degree)	Measured Length (In Pixel)	Measured Width (In Pixel)
90	90.00	74.00	19.00
75	75.57	75.38	18.4
60	52.34	77.05	15.83
45	44.18	76.05	15.33
30	32.94	77.24	13.05
15	3.65	62.90	0.51

results, all boxes on the conveyor were assumed to be in a 90 degree orientation at all times. The only possible error allowed in the implementation was the shifting along the Y axis.

CHAPTER VII

COMPUTER PROGRAMS DESCRIPTION

This section discusses each of the input/output variables, subroutines, and the robot ACL program. A flow diagram showing the sequence of the program execution is presented in Figure 10. The source listing of the BASIC program that runs the physical implementation model is presented in Appendix A.

7.1 Input variables

The following are the input variables entered by the user:

N variable is used to store the number of different types of boxes. It can be any integer number greater than zero.

L array is used to store the lengths of the box types entered by the user. The input values must be in integer numbers.

W array is used to store integer numbers that represent the widths of the box types. The value in the **W** array must be smaller than the value in the **L** array for the same box type. Both **L** and **W** are one-dimensional arrays.

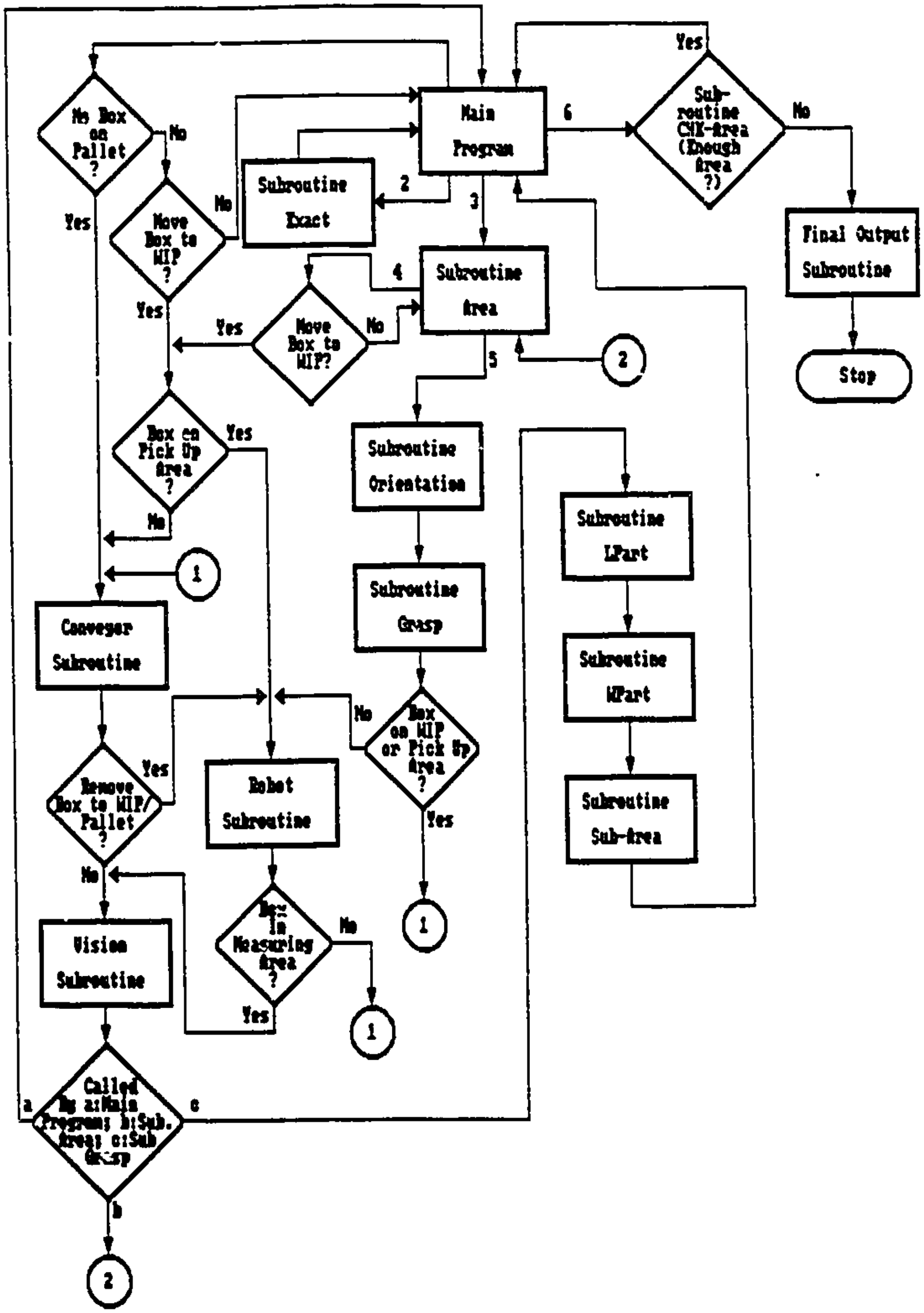


Figure 18. Interaction between subroutines

LENP is an integer value representing the length of the pallet in the model.

WIDP is an integer value representing the width of the pallet in the system.

7.2 Vision Subroutine (line 12000)

In order for the vision system to capture the image of a box and to display it on the monitor screen, a set of data statements must be read to set the address location which contains the picture to be displayed. Since this address location has to be set once for each complete cycle, it is not contained in the subroutine. Instead, it will be initialized at the beginning of the program. The function, **PEEK**, is used to store the numerical values of the digital signal. These values must be divided by 16 to correspond with the 16-gray levels digitizer.

The Vision Subroutine begins by taking a picture of the box and displaying it on the screen. Once the picture is captured, the scanning process can begin. It starts from one point and moves across the screen until a black pixel is reached. To make sure this black pixel is indeed an edge of the image, the four pixels which follow are tested. If they are all black pixels, the first black pixel found is considered an edge and stored in a variable, **HON1**. The scanning continues along the same row until it reaches a white pixel, **EHON**. Then the scanning process is repeated with another starting point and results in another edge known as **HON2**. The pixel values of these two points are calculated and stored in **XHON1**, **YHON1**, **XHON2**, and **YHON2**, respectively. From these four values, the angle (**ANG**) of the box lying on the conveyor is calculated by the following equation;

$$\mathbf{ANG} = \mathrm{Tan}^{-1}\left(\frac{\mathbf{YHON2 - YHON1}}{\mathbf{(XHON2 - XHON1)}*2.5725}\right)*180/\pi$$

Since the pixels are rectangular blocks, one pixel length is not equal to one pixel width. Therefore, the base, **XHON2 - XHON1**, must be multiplied by the ratic value 2.5725. Since, BASIC language provides only radian values for calculated angles, the angle must be multiplied by 180 degrees and divided by π.

Then, the mid-point, **MID**, between **HON1** and **EHON** is determined. The subroutine will find the length and width of the box by scanning up and down from **MID** and across the screen from **HON1**. The scanning follows the line generated by **HON1** and **HON2**. The four pixels are determined: **TOPL & BOTL**, as the top and the bottom pixels on the width of the box, respectively, and **RIGW & LEFW** as the right and left pixels on the length of the box, respectively. By taking the angle of the box into consideration, the length (**LENGTH**), the width (**WID**), and the area (**AREA**) of the box in pixels are calculated by the following equations;

$$\mathbf{LENGTH} = \frac{\mathbf{BOTL - TOPL} - 1}{\mathrm{Sin}(|(\mathbf{ANG}*\pi)/180|)}$$

$$\mathbf{WID} = (\mathbf{RIGW - LEFW}) * \mathrm{Sin}(|(\mathbf{ANG}*\pi)/180|)$$

$$\mathbf{AREA} = \mathbf{LENGTH} * \mathbf{WID}$$

In order to convert pixels to cm, two ratios are determined: a ratio between one cm and one pixel length, and another ratio between one cm and one pixel width. In the physical implementation, one pixel length equals 0.0706 cm and one pixel width equals 0.1756 cm. However, these two ratio values are only approximations and error factor must be taken into consideration when setting the set tolerance. In addition, the shifting of a box along the Y axis is determined. This is obtained by, setting a reference point, **CENTERY**, for the boxes. Then, the program calculates the Y value of the center point, **CENY**, of the box by the following equation;

$$\mathbf{CENY} = \frac{\mathbf{BOTL + TOPL}}{2}$$

Finally, **CENY** and **CENTERY** are compared. If both are equal, no shifting has occurred. Otherwise, the difference between them indicates the direction and the number of pixels that the box has shifted along the Y axis.

Once the dimensions of the box are measured, the box type can be determined by comparing the dimensions to the data variables with the set tolerance. Once all of the results are determined, they are shown on the screen.

7.3 Conveyor Subroutine (line 14000)

The Conveyor Subroutine controls the moving and the stoppage of the conveyor and sends signals to either the Vision Subroutine or the Robot Subroutine. The motion of the conveyor is controlled by a motor variable, **MOT1**, and a speed variable, S, in the subroutine. **MOT1** represents the motor number which can be any integer number between 1 and 16, which indicates that the interface is capable of supporting up to 16 motors. S, on the other hand, ranges from -127 to +127. The 127 represents the maximum speed of the conveyor with the middle point 0 which indicates that the conveyor is stopped, and the + or - represents the speed direction, and 0 means the conveyor is stopped.

The two photo cells or switches are controlled by variables **SW1** and **SW2**. They also can be any integer number between 1 and 16, since the interface can be connected to a maximum of 16 photo cells and/or mechanical switches. For this particular study, **SW1** and **SW2** must be different numbers or, in other words, the switches have to be connected to different outlets of the interface. The output signals, **A1** and **A2**, have values of 0 or 1, with 0 represents the photo cell or the switch as open, and 1 means that either the photo cell is blocked from the light source or the switch is triggered.

7.4 Robot Subroutine (line 15000)

The Robot Subroutine, or the transmission subroutine, is used to transmit the pick up and the target coordinates of a box to the robot controller and to run the ACL program to palletize the box. The process involved include opening the communication port as #1; printing ACL command statements to the port; and transmitting these command statements to the controller. After transmitting a command to the controller, Subroutine Wait is activated to delay any other command. At the end of this subroutine, the communication port must be closed. Otherwise, input/output device errors will occur.

7.4.1 Subroutine Open (line 16480): This subroutine is used to open the communication port between the computer and the robot controller.

7.4.2 Subroutine Wait (line 16580): The purpose of this subroutine is to check the controller to see whether it is free for accepting another transmission. If not, the subroutine goes into a loop until the controller is clear. This step is very important because if the controller is not free and another command statement is being transmitted to it, the latter statement might be lost and might not be recovered.

7.5 Main program and pallet subroutines

Part of the main program and eight subroutines are used to decide which box to be loaded on to the pallet and which box should go to the WIP area. A three-dimensional array called **GRID** is used to symbolize the pallet. If there is no box on the pallet, all values of the array are 1. Once a box is loaded on to the pallet, the corresponding locations in the array will become 0.

7.5.1 Main program

The first part of the Main Program is the initialization process. It includes the following operations:

1. set maximum dimensions for all arrays
2. allow the user to input data
3. set location coordinates on the WIP area

4. make sure that all motors are stopped at the beginning of the operation
5. initialize the **GRID** array by setting all elements to correspond to the first layer of the pallet to 1 and the rest to 0
6. set the memory segment for the Vision Subroutine.

The second part involves the determination of whether a box from the WIP area or the conveyor should be chosen for loading on to the pallet, or whether the front box on the conveyor should go to the WIP area and allow the box behind to move to the pick up area. If a box is required to be loaded on the WIP area, the program will store its type and orientation in one-dimensional arrays, **WIP**, **SWIPX**, **SWIPY**, **SWIPZ**, **SWIPP**, **SWIPR**, for later use. The first array stores the box type and the last five arrays represent the coordinates of the WIP location, which are the five degrees of freedom of the robot arm. Once a box is chosen to be loaded on to the pallet, the pallet subroutines will be activated to carry out their functions.

7.5.2 Pallet subroutines

The following are subroutines that carry out operations such as selecting the sub-area, determining the orientation of a box on the pallet, and partitioning the remaining area of the pallet.

7.5.2.1 Subroutine Area (line 2000): This subroutine is used for choosing the proper sub-area for loading the selected box. This is done by checking all sub-areas remaining on the lower layer and choosing the smallest one which is large enough to be the designated area. If none of the sub-areas is large enough in the lower layer, the subroutine checks the upper layer until this area is found. If the area is not large enough for the selected box, the subroutine chooses a smaller box from the WIP area provided there is a smaller box in the area. Otherwise, the subroutine will test another box on the conveyor.

7.5.2.2 Subroutine Orientation (line 4000) : Once the Subroutine Area is executed, the box is selected and the sub-area is determined. Subroutine Orientation is then called to determine the orientation of the box on the sub-area and to provide the pick up and target coordinates for the robot. This subroutine also sends these coordinates to the Robot Subroutine. Then, the elements in the **GRID** array which correspond to the location of the sub-area are changed to 0 so that no box can be loaded on to this area of the pallet again. In the meantime, the upper layer of that location becomes 1 in the **GRID** array to allow the loading of another box.

7.5.2.3 Subroutine Grasp (line 4500): Subroutine Grasp provides the orientation of the robot gripper. The robot can pick up a box either by the width side or by the length side. In order to reduce the interference between the gripper and the boxes on the pallet.

7.5.2.4 Subroutine LPart (line 5000): After a box is loaded on to the pallet, the remaining area is partitioned into rectangular areas. In the program itself, it is the **GRID** array that is being partitioned. There are two methods for partitioning this array. The first is the lengthwise partition and is carried out in Subroutine LPart. Basically, it checks each element against the bigger value in the array until it finds the first spot of an empty area. Then it moves across and up to find the upper right hand corner of this empty area. Once an empty area is found, its lower left hand corner coordinates, **TSTL & TSTW**, and upper right hand corner coordinates, **TFINL & TFINW**, will store in four two-dimensional arrays. This process continues until all empty space on one layer is checked. Then the subroutine goes to the second and/or the third layers and performs the same functions over again.

7.5.2.5 Subroutine CHK-Area (line 7000): This subroutine is for checking whether there is any more sub-area for even the smallest box in the system. If not, it calls up the Final Output Subroutine for printing out the final output.

7.5.2.6 Subroutine Exact (line 7500): Subroutine Exact is used to check if any of the sub-areas have the exact dimension for the box being tested. If so, the subroutine notifies the program to select this box for loading on to the pallet.

7.5.2.7 Subroutine WPart (line 8000): This subroutine performs the same function as Subroutine LPart but in this case, it is the widthwise partition. That is, the checking begins against the smaller value in the GRID array.

The four two-dimensional arrays used for storing the lower left hand and the upper right hand corner coordinates are **USTL** & **USTW**, and **UFINL** & **UFINW**, respectively.

7.5.2.8 Subroutine Sub-Area (line 10000): After the two partitions are completed, Subroutine Sub-Area is called up to decide which partition is better. This is done by comparing the largest sub-areas resulting from the two methods. The largest one represents the correct partition of the pallet. If a tie exists, the choice is arbitrary.

7.6 Final Output Subroutine (line 20000)

When Subroutine CHK-Area shows that that there are no more areas for any box type in the system, the palletization is completed and this subroutine is called upon to print out the results. The output includes the pattern of boxes loaded on to the pallet (**BOXES**); the types of boxes contained in the maximum WIP (**MAXWIPS**); the pallet utilization in percent (**PERCENT**); and the total palletization time in seconds (**TTIME$**). **BOXES** and **MAXWIPS** are one-dimensional arrays and **PERCENT** and **TTIME$** are variables.

7.7 ACL robot program

The ACL robot program is stored in the robot controller and is activated by the command "RUN ROBOT" which is sent to the controller by the Robot Subroutine. It manages all the material handling operations. The listing of the program can be seen in Appendix B.

The program always ensure that the gripper is open when it picks up a box. The robot arm move to the pick up location, which can be either on the conveyor or in the WIP area, to grasp the box. The position variable for the pick up location is **PICKU**. Once the box is grasped, the robot arm moves it to the destined sub-area and unloads it. The position variable for this unload location is **ULOAD**. Two other position variables are used in the program. They are **PIUP** and **UNLD**. The purpose of using these two position variables is to make sure the robot gripper moves right above **PICKU** and **ULOAD** before dropping to pick up or release the box, respectively. If these two points are not being used when the robot moves to **PICKU** or **ULOAD**, it moves on an angle and may crash into other boxes in the WIP area or on the pallet. These four position variables actually consist of five values each which are the five degrees of freedom of the robot arm.

The robot must pass through a dummy point which is either **DUM1** or **DUM2** depending on the movement. This dummy point allows the robot arm to move on a circular path to the destined position. In this way, the robot arm are prevented from crashing into other boxes or obstacles during the operation. If the dummy point is not used in this model, the robot interferes with other boxes being transported to the pallet or the WIP area.

Finally, since the robot controller is a multi-tasking system, it can handle different operations at the same time. Therefore, for minimizing the material handling errors, a DELAY command is used to make sure the controller does not run another command before the previous one is completed.

CHAPTER VIII

ANALYSIS OF RESULTS

In the last chapter, a series of subroutines and the final output from the program have been discussed. The first part of this chapter will analyze the results of the physical implementation. The second part will compare these results with others that result from existing models.

8.1 Illustration of results

The values of the output variables recorded at different iterations during one run of the program are shown in Appendix C. The results include the output from all the iterations and the final output at the end of the palletization. There are two types of output in each of the iterations. The first type is from the vision system and the second is from the algorithm. The following sections will discuss these two types of results and also the final output.

8.1.1 Vision system output

The vision system output consists of the image of a box, its dimensions and its orientation measured by the Vision Subroutine. In the first iteration, the picture shows an image of a type 3 box. The measurement, as shown in Appendix C, are not accurate. This is because the vision system is not capable of giving exact measurement. But after taking the set tolerance into consideration, the box type can be identified as a type 3, since the measured length, 4.8,lies between 4.6 cm and 5.4 cm, and the measured width, 2.9, is within the range of 2.6 cm and 3.4 cm. The angle shows that the box lies at a 90° angle so that no adjustment is required. However, a small shift has been recorded. The 2 pixels indicate that the robot gripper has to move 1 mm in a the positive direction along the Y axis in order to pick up the box at the center.

The second iteration is similar to the first one, except that the box is now identified as a type 2 box. Once again, the results indicate that no angle has been detected but shifting of 0.7 mm in the negative Y axis, has occurred.

8.1.2 Output from the algorithm

The output indicates the location of the box on the pallet, and the partition of the remaining empty space on the pallet after each palletization. In the first iteration, since the box is the first one that came into the system, no comparison can be made and no palletization is carried out.

Therefore, no information except for the vision output is available. The output is generated in the second iteration. The first algorithm output shows there is a type 3 box on the WIP area. This is because after the first two boxes have been measured, the algorithm chooses the bigger box, the second one, to load on to the pallet. However, the smaller box, the first one, is already occupying the pick up area. Therefore, the robot must move the smaller box to the WIP area.

At the beginning when there is no box on the pallet, there is only one sub-area and it is the dimension of the entire pallet. That is why the output indicates that the type 2 box, the larger of the first two boxes, is loaded on to area 1 on the first layer of the pallet. The bottom left hand corner and the top right hand corner coordinates are shown on the monitor screen and as expected, this area has coordinates (1,1) and (12,10), respectively.

The partition operation of the remaining area on the pallet follows the palletization. After the two methods of partitioning are applied and the comparison between the results of the two methods is completed, the program chooses the alternative which provides the largest sub-area. In this iteration, a tie exists and the program makes a decision about selecting the partition from the lengthwise method because its largest sub-area has the exact dimension of a type 1 box. This method separates the remaining area on the first layer of the pallet into two sub-areas. The first sub-area has the bottom and the top corner coordinates as (7,1) and (12,10), respectively. The length and the width of this sub-area, therefore, equals 10 cm and 6 cm, respectively. On the other hand, the second sub-area has the bottom corner coordinates as (1,6) and the top as (6,10) and its length and width are 6 cm and 5 cm, respectively. Moreover, a sub-area on the second layer becomes available when the type 2 box is loaded on to layer 1 of the pallet. Thus, this sub-area will have the same dimension as the type 2 box and its location is bounded by the two corner coordinates (1,1) and (6,5). The other iterations follow the same procedures.

8.1.3 Final output

When the pallet is completely filled, a set of final outputs are shown on the screen. These results are shown, following the last iteration output in Appendix C. The first part of the final output shows the pattern of the boxes that have been loaded on to the pallet during this particular run as 2-2-2-3-3-2-1-2-1-3-2-3. Next, the maximum WIP required, as indicated in the output, consists of only one type 3 box which means the total WIP area needed for this particular sequence has dimensions which measure 5 cm by 3 cm. The last two outputs, along with the maximum WIP, are considered the most important performance measures because they are the main objectives of the first algorithm. They are the pallet utilization and the total palletization time. As indicated in the result, the example has achieved a 100% pallet utilization and it took 5 minutes and 54 seconds for palletizing one pallet with this particular sequence. Other performance measures can be developed, such as the utilization of the robot, conveyor and vision systems.

8.2 Comparisons between proposed and existing methods

In order to validate the proposed algorithms, the results are compared with other results from existing heuristics. However, only the first algorithm is used in the comparison since the results can be compared to existing results which are provided in the literature. Although Hodgson [4] has briefly discussed the other two palletization methods, his methods, at that time, were still under experiment and had no concrete results. Since then, no further attempts have been made. The first algorithm is compared with the methods of Tanchoco and Penington [7], and Hodgson [4]. Since T&P described the step by step procedures of their algorithm, they have made it available for others to simulate the algorithm with different example sequences and to obtain the same kinds of performance measures as the physical implementation model. Therefore, the first comparison is based on these measures. The second comparison is different because Hodgson did not make his algorithm available to others. However, he did provide information on the CPU time for running his model and the pallet utilization when different box sizes are applied. These results will be considered in the second comparison.

8.2.1 Comparison with T&P's algorithm [7]

After running the physical implementation model with example sequences, the actual time for each individual operation is measured. These operations and their time, as shown in Table 9.

Table 9. Processing time required for individual operation

DESCRIPTIONS	TIME REQUIRED (SEC)
Robot moves between Conveyor and Holding Area (Loaded or empty)	31
Robot moves between Holding Area and Pallet (Loaded or empty)	31
Robot moves between Conveyor and Pallet (Loaded or empty)	33
Conveyor transports box from Loading area to measuring area	10
Conveyor transports box from measuring area to pick-up area	10
Vision system analysis time	19

The operations for T&P's algorithm are different from the proposed algorithm since the algorithm requires the accumulation of boxes in different stages. Therefore, other than the basic operations, additional movements are introduced to the robot. The added robot movements include travelling between the stage and the pallet, the holding area and the stage, and between stages. Thus, the travelling time required for the robot movements is changed. These changes are shown in Table 11.

Twenty sequences were used for this comparison. The number of box types has been increased from three, in the physical implementation, to four. The added box type is assigned as type 4 with dimensions of 12 inches by 10 inches. The total palletization time measured for each sequence is based on the procedures of the two algorithms and the types of boxes contained in the random sequence. Since some of these operations are carried out simultaneously at some point during the operation, their times are overlapping and only one of them is counted. The entire output resulting from both the proposed algorithm and T&P's algorithm is shown in Table 10 and Table 11, respectively. The output results include the total WIP, the maximum WIP and the pallet utilization.

The comparison between these two algorithms focuses on three performance measures. They are the total palletization time, the maximum WIP, and the pallet utilization. First of all, the total palletization time indicates that the proposed algorithm provides better results than T&P's algorithm in this performance measure. It is quite obvious that the stages and the grouping technique used in

Table 10. Summary of examples using proposed algorithm

NO.	SEQUENCE	TIME (SEC)	WIP REQUIRED (SQ. INCHES)	MAXIMUM WIP (SQ. INCHES)	UTILIZATION (%)
1	2-3-2-3-3-3-1-1-2-3-3-1	481	1(240)	1(240)	100
2	2-3-4-1-4-2-3-2-2-1-4-1-3	572	2(120),1(240),1(480)	1(120),1(480)	100
3	1-2-4-2-1-1-2-1-2	351	1(120),1(480)	1(120),1(480)	100
4	4-1-2-3-2-2-4-2-1-3-3-1	514	1(120),1(480)	1(480)	100
5	1-1-3-2-2-1-2-3-2-3-3	450	1(240)	1(240)	100
6	3-1-3-3-4-3-3-3-1-1-3-3-3-2-1-3	611	1(120),1(240),1(960)	1(120),1(960)	100
7	3-2-1-1-2-3-2-1-1	384	1(240)	1(240)	100
8	4-4-1-3-2-3-4-3-2-3-4-3-2-4-1-3-4-2-4	799	3(120),1(480),1(960)	1(480),1(960)	98
9	3-2-1-2-2-1-3-4-1-1	415	1(120),1(240)	1(240)	100
10	4-1-4-1-2-1-4-1-1-2-4-3-3	541	2(120),1(480),1(960)	1(960)	100
11	3-2-2-3-2-3-3-3-4-4-1-3-1-1	576	3(240)	1(240)	100
12	1-4-2-4-2-4-4-1-2-3-3-1-3-4-2	613	1(120),1(240),1(480)	1(480)	100
13	4-1-3-2-4-3-2-4-4-2-2-3-2-1-2	609	1(120),2(240)	1(240)	100
14	2-3-3-3-4-3-3-3-4-4-4-4-4-4-3-3-1-1-3-3-4	830	2(120),1(960)	1,(120),1(960)	100
15	2-3-3-3-2-1-3-3-3-4-3-1-2-2-4	640	1(120),1(240),1(480)	1(480)	100
16	3-2-3-3-4-3-3-1-3-1-1-1	510	1(120),2(240)	1(120),1(240)	100
17	2-4-2-1-1-4-4-3-4-4-1-4-3-3-1-3-3	704	3(120),1(960)	1(960)	100
18	2-4-3-3-2-3-2-4-3-4-2-3-1-2-4-3-4-4-4-4	776	2(120)	1(120)	100
19	2-3-3-1-1-4-1-1-1	382	1(120),1(240)	1(240)	100
20	4-1-3-3-2-3-3-2-4-4-4-1-2-3-3-2	673	1(120),2(240)	1(120,1(240)	100

box types : 1 - 40 in. * 24 in. (960 in^2) ; 2 - 24 in. * 20 in. (480 in^2)
3 - 20 in. * 12 in. (240 in^2) ; 4 - 12 in. * 10 in. (120 in^2)

Table 11. Results of examples using T&P's algorithm
(KEYS : H.A. - Holding Area ; STA. - STAge)

NO.	SEQUENCE	TIME (SEC)	WIP (SQ. INCHES)	UTILIZATION (%)
1	2-3-2-3-3-3-1-1-2-3-3-1	691	H.A. 6(240),3(480),3(960) STA. 2(240)	100
2	2-3-4-1-4-2-3-2-2-1-4-1-3	720	H.A. 3(120),3(240),4(480),3(960) STA. 2(120),2(240)	100
3	1-2-4-2-1-1-2-1-2	500	H.A. 1(120),4(480),4(960) STA. 2(480)	100
4	4-1-2-3-2-2-4-2-1-3-3-1	690	H.A. 2(120),3(240),4(480),3(960) STA. 2(120),2(240)	100
5	1-1-3-2-2-1-2-3-2-3-3	632	H.A. 4(240),4(480),3(960) STA. 2(240),2(480)	100
6	3-1-3-3-4-3-3-3-1-1-3-3-3-2-1-3	869	H.A. 1(120),10(240),1(480),4(960) STA. 2(240),2(480)	100
7	3-2-1-1-2-3-2-1-1	530	H.A. 2(240),3(480),4(950) STA. 2(240),2(430)	100
8	4-4-1-3-2-3-4-3-2-3-4-3-2-4-1-3-4 -2-4	999	H.A. 7(120),6(240),4(480),2(960) STA. 2(120),2(240),2(480)	100
9	3-2-1-2-2-1-3-4-1-1	559	H.A. 1(120),2(240),3(480),4(960) STA. 2(240),3(480)	100
10	4-1-4-1-2-1-4-1-1-2-4-3-3	737	H.A. 4(120),2(240),2(480),4(960) STA. 2(120),2(240),2(480)	100
11	3-2-2-3-2-3-3-3-4-4-1-3-1-1	779	H.A. 2(120),6(240),3(480),3(960) STA. 2(120),2(240),2(480)	100
12	1-4-2-4-2-4-4-1-2-3-3-1-3-4-2	808	H.A. 5(120),3(240),4(480),3(960) STA. 2(120),2(240),2(480)	100
13	4-1-3-2-4-3-2-4-4-2-2-3-2-1-2	822	H.A. 4(120),3(240),6(480),2(960) STA. 2(120),2(240),2(480)	100
14	2-3-3-3-4-3-3-3-4-4-4-4-4-4-3-3 1-1-3-3-4	1186	H.A. 8(120),10(240),1(480),2(960) STA. 2(120),2(240),2(480)	100
15	2-3-3-3-2-1-3-3-3-4-3-1-2-2-4	838	H.A. 2(120),7(240),4(480),2(960) STA. 2(120),2(240),2(480)	100
16	3-2-3-3-4-3-3-1-3-1-1-1	666	H.A. 1(120),6(240),1(480),4(960) STA. 2(240),2(480)	100
17	2-4-2-1-1-4-4-3-4-4-1-4-3-3-1-3 -3	940	H.A. 6(120),5(240),2(480),4(960) STA. 2(120),2(240),2(480)	100
18	2-4-3-3-2-3-2-4-3-4-2-3-1-2-4-3 -4-4-4-4	1112	H.A. 8(120),6(240),5(480),1(960) STA. 2(120),2(240),2(480)	100
19	2-3-3-1-1-4-1-1-1	508	H.A. 1(120),2(240),1(480),5(960) STA. 2(240),2(480)	100
20	4-1-3-3-2-3-3-2-4-4-4-1-2-3-3-2	896	H.A. 4(120),6(240),4(480),2(960) STA. 2(120),2(240),2(480)	100

Robot movement time :
Between conveyor and holding area (loaded or empty) = 15 sec
Between holding area and stage (loaded or empty) = 15 sec
Between stages (loaded or empty) = 15 sec
Between stage and pallet (loaded or empty) = 15 sec
Between holding area and pallet (loaded or empty) = 31 sec

Conveyor transportation time :
From loading point to measuring point = 10 sec
From measuring point to pick-up point = 10 sec

Vision system analysis time = 19 sec

box type : 1 - 40 in. * 24 in. (960 in^2) ; 2 - 24 in. * 20 in. (480 in^2)
3 - 20 in. * 12 in. (240 in^2) ; 4 - 12 in. * 10 in. (120 in^2)

T&P's algorithm increase the palletization time. Although most of the robot movement time using T&P's algorithm is about half of that required in the proposed algorithm. But the grouping technique employed in T&P's algorithm increases the number of movements by the robot. This, in turn, increases the total palletization time for filling up a pallet.

The situation becomes worse when the sequence includes a large number of boxes which require the grouping technique. These boxes are type 2, 3, and 4. This can be justified by taking sequences #3, #14, #18 and #19 as examples. In sequences #3 and #19, the number of boxes requiring the grouping technique are five and four, respectively. The total palletization times saved by the proposed algorithm are 149 and 126 seconds in sequence #3 and #19, respectively (i.e. 42.45 % and 32.98 %, respectively). On the other hand, sequences #14 and #18 contain a total of 38 boxes that require the grouping technique and a majority of them have to go through at least two stages before loading on to the pallet. As a result, the saving in palletization time resulting from the proposed algorithm applied to sequence #3 and #19 are 348 and 336 seconds, respectively (i.e. 41.53 % and 43.30 %, respectively).

This grouping technique also explains why the area required for the WIP is so large in T&P's algorithm. Although T&P mentioned that their algorithm deals with the random incoming of boxes, this is not true. The boxes are placed first in a holding area, then the grouping technique is applied to group boxes into different stages before loading them on to the pallet. Thus the arrival of boxes is not random, at all. Therefore, the WIP area must be large enough to hold all boxes in each sequence and also reserve space for different stages. Also, the size of the holding area is dependent on the box types in the sequence as is the total area for the stages. The stages are used for all but the type 1 boxes. If there are more than two of the same box types contained in the sequence, a stage with dimension equaling two of this box type is required. Also, when one type of box requires the use of a stage, stages for other box types larger than this must be added to the work area even though there is only one box from each of the other box types contained in the sequence. This is because small boxes must go through their own stage as well as the stages for larger boxes in order to be loaded on to the pallet.

For example, sequence #19 has two stages, one for type 3 boxes and the other for type 2 boxes. Since there is only one type 4 box in the sequence, a stage for this box type is unnecessary. The two type 3 boxes satisfy the basic requirement for a stage in the work area. Once this stage is added, it automatically forces the installation of a stage for the type 2 box even though this type has only one box in the sequence. In T&P's algorithm, the robot does not load any boxes from the stage on to the pallet unless the total area of the boxes on the stage equals the size of a type 1 box.

As in the proposed algorithm, the WIP area is used only when necessary. And when a box is moved from the WIP area to the pallet, that area will again be available for any other boxes moved to the WIP area. Therefore, the dimension of the holding area is based only on the maximum WIP for each individual sequence. To indicate the differences between the WIP area required in the two algorithms, sequence #19 is once again taken as an example. The sequence consists of one type 4 box, two type 3 boxes, one type 2 box, and five type 1 boxes. As indicated earlier, this sequence requires two stages, one for the type 3 boxes and the other for the type 2 box in T&P's method. The total, or the maximum, WIP area required by T&P's algorithm is the holding area of all these boxes plus the area for these two stages. The total holding area in this case equals 5880 sq. inches and the stages take up 1440 sq. inches. The maximum WIP area required in T&P's algorithm, therefore, equals 7320 sq. inches. For the proposed algorithm, the maximum WIP area is only equal to one type 3 box or 240 sq. inches, a saving of 2950 %.

The size of the WIP area in T&P's algorithm is directly proportional to the number of boxes in the sequence, that is, the bigger the number of boxes in a sequence, the larger the WIP area required. For the proposed algorithm, the maximum WIP area depends more on the pattern of boxes in the sequence than on the number of boxes. Thus, the proposed algorithm provides significant savings in WIP area.

The last measure of comparison is the pallet utilization. Since T&P's algorithm waits until all boxes are in the holding area before any grouping and palletizing begins, it will always be capable of achieving 100 % utilization. On the other hand, the sequence for the proposed algorithm is truly randomly distributed and for the purpose of minimizing both the palletization time and the space used for the WIP, it does not wait for all boxes to come into the system before beginning the palletization process. In other words , it takes whatever is available on the conveyor and/or in the WIP area to begin the palletizing. Therefore, it may not achieve the optimal 100 % utilization. But this situation does not happen very often. In fact, out of the twenty sequences that were examined, only once, in sequence #8, the algorithm achieved an utilization of 98 %.

Based on the comparisons discussed, the proposed algorithm provides the better results of the two algorithms. Although T&P's algorithm has an advantage with achieving a consistent 100 % utilization over the proposed one, this advantage becomes less valuable if compared using other performance measures. For example, in sequence #8, where the proposed algorithm fails to achieve 100 % pallet utilization, the T&P algorithm provides a 2 % advantage in utilization over the proposed algorithm. However, the proposed algorithm has savings of 25.03 % (799 to 999 seconds) in palletization time and 441.67 % (1440 to 7880 sq. inches) in maximum WIP area.

Another disadvantage of T&P's algorithm is that the grouping technique is not applicable for all three-dimensional palletization problems with boxes of similar height. In both examples given in T&P's paper and in this research, one type of box always has an area equal to twice the size of the next type. For example, a type 1 box has an area equal to two type 2 boxes, a type 2 box is the same as two type 3 boxes, and so on. Therefore, if the system uses boxes of different dimensions, as in Hodgson's paper [4], the grouping technique will encounter serious problems. Also, the grouping technique is not a practical tool in the implementation of the palletization process. As the robot is picking up boxes in a group, variations in box dimensions may cause some of them to slip from the gripper. Therefore, based on these results, the proposed algorithm is more efficient and has better space utilization when compared to the algorithm resulting from T&P's research. In addition, it also guarantees more effective loading by the robot than the grouping technique employed by the T&P algorithm.

8.2.2 Comparison with Hodgson's approach [4]

Due to the fact that in his paper, Hodgson did not discuss the actual procedure of his palletizing algorithm, the only measures that have been provided are tables with computation times in virtual seconds and percent coverage, or utilization, of the pallet area. Therefore, these will be the performance measures used for comparison in this section. The complete result of the computation time and utilization percentage for both approaches is shown in Table 12.

Table 12. Summary of results by Hodgson's & proposed methods

Computation times in virtual seconds						
upper/lower bounds	20/20	18/22	17/23	16/24	15/25	10/30
Hodgson's approach	N/A	0.155	0.350	0.864	1.437	20.14
Proposed Algorithm	5.00	6.00	6.00	6.00	8.00	8.00
Percent coverage of the pallet area						
upper/lower bounds	20/20	18/22	17/23	16/24	15/25	10/30
Hodgson's approach	68.80	75.50	83.50	93.20	99.00	99.70
Proposed Algorithm	68.80	77.37	86.86	94.00	100.0	100.0

Because of the limited memory allowed in the BASIC compiler, not all of the examples discussed in Hodgson's paper can be examined here. The pallet dimension examined is 70 ft.by 50 ft. Even with this dimension, the memory limitation only enables the proposed algorithm to consider one layer on the pallet. The upper and lower bound of the box size indicates the range of the box dimensions. Six ranges of box dimensions, in feet, are examined. They are 20/20, 18/22, 16/24, 15/25, and 10/30 with the first number representing the upper bound and the second number representing the lower. For example, a box size with the upper bound of 22 and the lower bound of 18 includes boxes with dimensions of 18x18, 18x19, 18x20, 18x21, 18x22, 19x19, ..., 21x22, and 22x22, a total of 15 different box sizes. All of these boxes are assumed to be uniform in height.

The first comparison is between the computation times which only depend on the pallet size and the range of box dimensions in Hodgson's model. It is very difficult to compare the two approaches for the computation since each model uses different computer system. While Hodgson used an IBM 3033 with 64 bits main frame computer, the proposed algorithm employs a 386 microcomputer with 32 bits. When the upper and lower bounds are 18 and 22,respecively, Hodgson's approach only required 0.155 CPU seconds while the proposed algorithm takes 6 seconds to fill the pallet. However, the computation time in each case for the proposed algorithm still remains under the maximum limit of 30 seconds defined by Hodgson. In fact, as the range of box dimensions becomes larger, as in the case of 10/30, the proposed algorithm is even faster in generating solutions than Hodgson's approach (i.e. 8 seconds in the proposed algorithm as compared to 20.14 seconds in Hodgson's).

In the comparison of percent utilization of the pallet area, with all boxes having dimensions of 20x20, the best possible utilization would be only 68.6 %. As the range of the box dimensions increases, the percentage of utilization increases. With box dimensions randomly distributed between 15/25, and 10/30, the utilization achieved 100 % by the proposed algorithm. These results depend on particular sequences and do not occur all the time. Although the utilization resulted from the proposed algorithm applied to specific sequence as shown in Table 12, is better than Hodgson's, other sequences can result in different utilization value. Therefore, it is impossible to have a fair comparison between the two approaches. It would only be fair if both approaches dealt with the same sequences. But this cannot be done. Hodgson neither mentioned anything about the random sequences in his examples nor did he provide the step by step procedures of his approach for others to follow. The comparison, though, does serve the purpose that the proposed algorithm is capable of handling boxes of different dimensions. Also, the execution time by using the proposed algorithm, is smaller for large ranges between lower and upper bound of boxes. The pallet utilization, generated by the algorithm for the applied sequence is equally good or better than that of Hodgson.

CHAPTER IX

CONCLUSIONS

This study provides insight into three-dimensional palletization problems. The physical model provides an idea of the practicality of the first algorithm when it is implemented in a real world environment. With the help of this physical model, the implemented algorithm shows better results than those of existing methods. They include the palletization time and the space required for the WIP. The proposed algorithm and the robot sequence program are independent of the arrival pattern of boxes, the quantity of box types, the pallet dimensions, and the box dimensions. However, due to equipment problems, the model requires much more human intervention. But if all the hardware components work properly, this model can be a fully automated system.

Further modifications are made in the first algorithm and are reflected in the second and third algorithms. The changes and the additions of rules enhance the capabilities of the algorithms, such as palletizing boxes grouped by similar heights. The third and the last algorithm enables the user to deal with the worst possible case in the palletization problem whereby all boxes are different heights.

Although the proposed three algorithms have shown many improvements compared to existing algorithms and have expanded the horizon in dealing with the palletization problem, further studies can be investigated, particularly the following:

1. maximize the utilization of for the robot, conveyor, and vision system to reduce the idle time of each system.
2. develop computer programs to carry out the second and third algorithms.
3. implement these programs into physical models for simulation purposes and test the practicality of these systems.
4. selection of different patterns of boxes.
5. graphic animation of CAD.

REFERENCES

1. Tanchoco, J.M. and Agee, M.H. "Plan Unit Loads To Interact With All Components Of Warehouse System," Industrial Engineering, June 1981, pp. 36-48.

2. Smith, A. and DeCani, P. "An Algorithm To Optimize The Layout Of Boxes In Pallet," J. Operating Research Society, V. 31, 1980, pp. 573-578.

3. Kulick, A. "Interlocking Pallet Pattern Simulation Program," Industrial Engineering, September, 1982, pp. 22-24.

4. Hodgson, T.J. "A Combined Approach To The Pallet Loading Problem," IIE Trans., V. 14, No. 3, 1982, pp. 175-182.

5. Hodgson, T.J.; Hughes, D.S. and Martin-Vega, L.A. "A Note On A Combined Approach To The Pallet Loading Problem," IIE Trans., V. 15, 1983, pp. 268-271.

6. Puls, F.M. and Tanchoco, J.M.A. "Robotic Implementation Of Pallet Loading Patterns, Int. J. Production Research, V.24, No. 3, 1986, pp. 635-645.

7. Penington, R.A. and Tanchoco, J.M.A. "Robotic Palletization Of Multiple Box Sizes," Int. J. Production Research, V. 26, No. 1, 1988, pp. 95-105.

8. Steudel, H.J. "Generating Pallet Loading Patterns: A Special Case Of The Two-Dimensional Cutting Stock Problem," Management Science, V. 25, No. 10, 1979, pp. 997-1004.

9. Christofides, N. and Whitlock, C. "An Algorithm For Two-Dimensional Cutting Problems," Operations Research, V. 25, No. 1, 1977, pp. 30-44.

10. Löschau, G. "Stability Criteria For Column Stocks" Packaging Technology And Science, Vol. 2, 1989, pp. 155-163.

11. Lawrence, K.D. and Zanakis, S.H. "Production Planning And Scheduling: Mathematical Programming Applications" Industrial Engineering and Management Press, IIE., 1984, pp.1.

12. Eskenazi, R. "Video Signal Input" Robotics Age, Vol. 3, No. 2, March/April, 1981.

13. Cunningham, R. "Segmenting Binary Images", Robotics Age, Vol. 3, No. 4, July/August, 1981.

APPENDIX A

LISTING OF "THESIS.BAS"

```
10 'MAIN PROGRAM
15 DIM WIPX(4),WIPY(4),WIPZ(4),WIPP(4),WIPR(4),HEIGHT(3)
20 DIM SWIPX(10),SWIPY(10),SWIPZ(10),SWIPP(10),SWIPR(10)
25 DIM FWIPX(10),FWIPY(10),FWIPZ(10),FWIPP(10),FWIPR(10)
30 DIM L(5),W(5),LP(3,15),WP(3,15),WIP(10),GRID(3,13,11)
50 DIM STL(3,5),STW(3,5),FINL(3,5),FINW(3,5),MAXWIPS(10),BOXES(30)
70 DIM TSTL(3,5),TSTW(3,5),TFINL(3,5),TFINW(3,5),TWP(3,5),TLP(3,5)
90 DIM USTL(3,5),USTW(3,5),UFINL(3,5),UFINW(3,5),UWP(3,5),ULP(3,5)
91 DIM HIGH(3),MAX(3),MAX1(3),MAX2(3)
100 MAXWIP=0 : LEE=1
101 ' INPUT OF DATA
105 INPUT "Length of pallet :",LENP
106 INPUT " Width of pallet :",WIDP
107 INPUT "No. of box types ?",N
108 FOR I=1 TO N
109 PRINT "Length of type";I;"box ?"
110 INPUT L(I)
111 PRINT "Width of type ";I;"box ?"
112 INPUT W(I)
113 NEXT I
114 INPUT B$
120 HEIGHT(1)=0 : HEIGHT(2)=300 : HEIGHT(3)=600
130 LP(1,1)=LENP
150 WP(1,1)=WIDP
160 COUNT=0
170 AN1=1
171 OBOX=0
190 LAY=1
210 M=1
230 MMM=1
250 T=AN1
260 WW=4
261 TRUE=0
262 MOT1=1
263 SW1=10
264 SW2=12
```

```
270 STL(1,1)=1
280 STW(1,1)=1
290 FINL(1,1)=LENP
310 FINW(1,1)=WIDP
312 ' INITIALIZE LOCATIONS ON WIP
313 WIPX(1)=2723 : WIPY(1)=2937 : WIPZ(1)=290
314 WIPP(1)=-879 : WIPR(1)=-455
315 WIPX(2)=2723 : WIPY(2)=1977 : WIPZ(2)=290
316 WIPP(2)=-881 : WIPR(2)=-351
317 WIPX(3)=1916 : WIPY(3)=2972 : WIPZ(3)=290
318 WIPP(3)=-908 : WIPR(3)=-560
319 WIPX(4)=1916 : WIPY(4)=1750 : WIPZ(4)=290
320 WIPP(3)=-879 : WIPR(4)=-457
321 ' STOP ALL MOTORS
322 S=0
323 FOR I=1 TO 16
324      OUT 784-I,127+S
325 NEXT I
329 ' INITIALIZATION OF THE PALLET
330 FOR I = 1 TO FINL(1,1)
350      FOR J=1 TO FINW(1,1)
370              GRID(1,I,J)=1 : GRID(2,I,J)=0 : GRID(3,I,J)=0
390      NEXT J
410 NEXT I
560 ' SET UP SCREEN AND MEMORY SEGMENT FOR VISION SUBROUTINE
563 OUT &HA0,0
564 SCREEN 2
565 CLS
566 DATA &Hb8,&H00
567 '
568 'The following data statement sets the memory segment (*100h)
569 'which contains the picture to be displayed.
570 '
571 DATA &Ha0
572 DATA &H8e,&Hd8,&Hb8,&H00,&Hb8,&H8e,&Hc0,&Hbe,&H00,&H00,&Hbf
573 DATA &H00,&H00,&Hb2,&H00,&H8a,&H04,&H24,&H80,&H00,&Hc2,&Hd0
```

```
574 DATA &He8,&H00,&Hc2,&H8a,&H04,&H24,&H08,&Hd0,&He0,&H00,&Hc2
575 DATA &Hd0,&He0,&H00,&Hc2
576 DATA &H46,&H8a,&H04,&H24,&H80,&Hb1,&H04,&Hd2,&He8,&H00,&Hc2
577 DATA &Hd0,&He8,&H00,&Hc2,&H8a,&H04,&H24,&H08,&Hb1,&H02,&Hd2
578 DATA &He8,&H00,&Hc2,&Hd0,&He8,&H00,&Hc2,&H46,&H26,&H88,&H15
579 DATA &H47,&H81,&Hfe,&H00,&H63,&H74,&H17
580 DATA &H89,&Hf0,&H24,&H7f,&H75,&Hba,&H83,&Hef,&H40,&Hb8,&H00
581 DATA &H20,&H31,&Hc7,&H21,&Hf8,&H75,&H03,&H83,&Hc7,&H50,&Heb
582 DATA &Ha9,&H8c,&Hd0,&H8e,&Hd8,&H8e,&Hc0,&Hcb,&H90,&H90,&H90
583 DATA &H90,&H90
584 DEF SEG=&H3000                '}     USED TO
585 FOR A=0 TO 114                '}     PUT MACHINE
586 READ B                        '}     LANGUAGE SUBROUTINE
587 POKE A,B                      '}     INTO MEMORY
588 NEXT                          '}
598 ADDRESS=0             'MUST BE ZERO
599 CLS
600 EXAMINE=0
601 TIME$="00:00:00"
610 GOSUB 14000
650 FOR I=1 TO WW
670 IF WIP(I)=0 THEN GOTO 690 ELSE SS=I : GOTO 730
690 NEXT I
710 GOTO 1240
730 FOR I=SS TO WW
740 IF NBOX=99 THEN GOTO 745 ELSE GOTO 750
745 IF WIP(I) > 0 AND WIP(I) <= OBOX THEN GOTO 770 ELSE GOTO 1030
750 IF WIP(I) > 0 AND WIP(I) <= NBOX THEN GOTO 770 ELSE GOTO 1030
770 S=I
790 IF I+1 > WW THEN GOTO 870 ELSE 810
810 FOR J=I+1 TO WW
830 IF WIP(J)>0 AND WIP(J) < WIP(S) THEN S=J
850 NEXT J
870 BOX=WIP(S) :SX=SWIPX(S) : SY=SWIPY(S) : SZ=SWIPZ(S)
871 SP=SWIPP(S) : SR=SWIPR(S) : CY=0 : CR=0
880 EXAMINE=EXAMINE+1
```

```
890 GOSUB 2000
930 FOR K=1 TO WW
950      IF WIP(K)=0 THEN GOTO 990 ELSE GOTO 970
970      PRINT "WIP(";K;")=";WIP(K)
990 NEXT K
991 TTIME$=TIME$ : INPUT B$
992 TIME$=TTIME$
995 IF NBOX=99 AND OBOX>0 THEN GOTO 650 ELSE GOTO 1000
1000 IF NBOX=99 AND OBOX=0 THEN GOTO 610 ELSE GOTO 1010
1010 GOTO 1650
1030 NEXT I
1050 CHECK=0
1070 FOR SAGE=SS TO WW
1090      IF WIP(SAGE) > 0 THEN GOTO 1110 ELSE GOTO 1230
1110      TEMP=WIP(SAGE)
1130      GOSUB 7500
1150      IF CHECK=0 THEN GOTO 1230 ELSE GOTO 1170
1170      S=SAGE
1190      I=S
1210      GOTO 790
1230 NEXT SAGE
1240 IF NBOX=99 THEN TEMP=OBOX : GOTO 1270 ELSE GOTO 1250
1250 TEMP=NBOX
1270 CHECK=0
1290 GOSUB 7500
1310 IF CHECK=0 THEN GOTO 1330 ELSE GOTO 1350
1330 IF OBOX=0 THEN GOTO 1335 ELSE GOTO 1360
1335 FOR I=1 TO WW
1337     IF WIP(I)=0 THEN GOTO 1338 ELSE GOTO 1610
1338 NEXT I
1340 OBOX=NBOX : OSY=NSY : OSR=NSR
1342 GOSUB 14000
1344 GOTO 1370
1350 IF OBOX=0 THEN GOTO 1610 ELSE GOTO 1492
1360 IF NBOX=99 THEN GOTO 1395 ELSE GOTO 1370
1370 IF OBOX <= NBOX THEN GOTO 1390 ELSE GOTO 1460
```

```
1390 SX=4207 : SY=-908 : SZ=1326 : SP=-908 : SR=1006
1395 BOX=OBOX : CY=OSY : CR=OSR
1400 EXAMINE=EXAMINE+1
1410 GOSUB 2000
1450 GOTO 1650
1460 CHECK=0
1462 TEMP=OBOX
1464 GOSUB 7500
1466 IF CHECK=0 THEN GOTO 1492 ELSE GOTO 1390
1492 FOR K = 1 TO 4
1502    IF WIP(K)=0 THEN GOTO 1503 ELSE GOTO 1507
1503    SWIPX(K)=WIPX(K):SWIPY(K)=WIPY(K):SWIPZ(K)=WIPZ(K)
1504    SWIPP(K)=WIPP(K):SWIPR(K)=WIPR(K):WIP(K)=OBOX
1505    FX=WIPX(K):FY=WIPY(K):FZ=WIPZ(K):FP=WIPP(K):FR=WIPR(K)
1506    CY=OSY:CR=OSR:GOTO 1508
1507 NEXT K
1508 SX=4307 : SY=-908 : SZ=1326 : SP=-908 : SR=1006
1510 GOSUB 15000
1515 OBOX=0 :OSY=0 : OSR=0
1520 TOTWIP=0
1530 FOR K=1 TO WW
1550    IF WIP(K)=0 THEN GOTO 1590 ELSE GOTO 1570
1570    PRINT "WIP(";K;")=";WIP(K)
1580    TOTWIP=TOTWIP + (W(WIP(K)*L(WIP(K)))
1590 NEXT K
1591 TTIME$=TIME$ : INPUT B$
1592 TIME$=TTIME$
1593 IF TOTWIP>MAXWIP THEN GOTO 1594 ELSE GOTO 1610
1594 MAXWIP=TOTWIP : EDRIC=1
1595 FOR K=1 TO WW
1596    IF WIP(K)=0 THEN GOTO 1598 ELSE GOTO 1597
1597    MAXWIPS(EDRIC)=WIP(K) : EDRIC=EDRIC + 1
1598 NEXT K
1610 BOX=NBOX : CY=NSY : CR=NSR
1612 GOSUB 2000
1614 GOSUB 7000
```

```
1616 GOTO 650
1650 GOSUB 7000
1655 OBOX=NBOX : OSY=NSY : OSR=NSR
1670 GOTO 610
2000 '
2020 '
2040 'SUBROUTINE AREA
2060 LAYER=LAY
2080 MMM=1
2090 HT=HEIGHT(LAY)
2100 IF FIX(LP(LAYER,MMM)/L(BOX))*FIX(WP(LAYER,MMM)/W(BOX))>0
        OR FIX(LP(LAYER,MMM)/W(BOX))*FIX(WP(LAYER,MMM)/L(BOX))>0
        THEN GOTO 2120 ELSE GOTO 2180
2120 MINL=LAYER
2140 MINA=MMM
2145 IF BOX=WIP(S) THEN Z=S
2160 GOTO 3040
2180 MMM=MMM+1
2200 IF MMM > T THEN GOTO 2220 ELSE GOTO 2100
2220 LAYER=LAYER+1
2240 IF LAYER=2 THEN GOTO 2250 ELSE GOTO 2320
2250 IF LAY=1 THEN GOTO 2251 ELSE HT=HEIGHT(2) : GOTO 2260
2251 HT=HEIGHT(1)+300 : GOTO 2260
2260 MMM=1
2280 T=AN2
2300 GOTO 2100
2320 IF LAYER=3 THEN GOTO 2330 ELSE GOTO 2400
2330 IF LAY=1 THEN HT=HEIGHT(1)+600
2332 IF LAY=2 THEN HT=HEIGHT(2)+300
2334 IF LAY=3 THEN HT=HEIGHT(3)
2340 MMM=1
2360 T=AN3
2380 GOTO 2100
2400 IF BOX=WIP(S) THEN GOTO 2420 ELSE GOTO 2502
2420 IF OBOX > 0 THEN GOTO 2422 ELSE GOTO 2424
2422 IF OBOX > BOX THEN GOTO 2440 ELSE GOTO 2424
```

```
2424 IF NOBX > BOX THEN GOTO 2426 ELSE GOTO 2502
2426 BOX=NBOX : CY=NSY : CR=NSR
2428 GOTO 2460
2440 BOX=OBOX : CY=OSY : CR=OSR
2460 LAYER=LAY
2480 IF LAYER=1 THEN T=TAN1 : GOTO 2080 ELSE GOTO 2240
2502 SX=4307 : SY=-908 : SZ=1326 : SP=-908 : SR=1006
2507 FOR K = 1 TO 4
2508    IF WIP(K)=0 THEN GOTO 2510 ELSE GOTO 2514
2510    FX=WIPX(K):FY=WIPY(K):FZ=WIPZ(K):FP=WIPP(K):FR=WIPR(K)
2511    SWIPX(K)=WIPX(K):SWIPY(K)=WIPY(K):SWIPZ(K)=WIPZ(K)
2512    SWIPP(K)=WIPP(K):SWIPR(K)=WIPR(K):Q=K
2513    GOTO 2515
2514 NEXT K
2515 IF OBOX>0 THEN WIP(Q)=OBOX:CY=OSY:CR=OSR:GOTO 2525
2517 WIP(Q)=NBOX : TRUE=1 : CY=NSY : CR=NSR
2519 GOSUB 14000
2521 GOTO 2530
2525 GOSUB 15000
2530 TOTWIP=0
2540 FOR K=1 TO WW
2560    IF WIP(K)=0 THEN GOTO 2600 ELSE GOTO 2580
2580    PRINT "WIP(";K;")=";WIP(K)
2590    TOTWIP=TOTWIP + (W(WIP(K))*L(WIP(K)))
2600 NEXT K
2601 TTIME$=TIME$ : INPUT B$
2602 TIME$=TTIME$
2605 IF TOTWIP>MAXWIP THEN GOTO 2607 ELSE GOTO 2630
2607 MAXWIP=TOTWIP : EDRIC=1
2610 FOR K=1 TO WW
2615    IF WIP(K)=0 THEN GOTO 2625 ELSE GOTO 2620
2620    MAXWIPS(EDRIC)=WIP(K) : EDRIC = EDRIC + 1
2625 NEXT K
2630 LARGE=BOX
2640 FOR I=1 TO WW
2660    IF WIP(I) > LARGE GOTO 2662 ELSE GOTO 2780
```

```
2662 LAR=WIP(I)
2680    IF I+1 > WW THEN GOTO 2682 ELSE GOTO 2700
2682    LARGE=WIP(I):BOX=WIP(I):SX=SWIPX(I):SY=SWIPY(I)
2684    SZ=SWIPZ(I):SP=SWIPP(I):SR=SWIPR(I)
2686    SS=I:CY=0:CR=0:GOTO 2840
2700    FOR J=I+1 TO WW
2720            IF WIP(J)<=LAR AND WIP(J)>LARGE
                   THEN LAR=WIP(J):SS=J:GOTO 2740
                   ELSE GOTO 2740
2740    NEXT J
2760    GOTO 2820
2780 NEXT I
2800 GOTO 3220
2820 LARGE=LAR:BOX=LAR:SX=SWIPX(SS):SY=SWIPY(SS)
2830 SZ=SWIPZ(SS):SP=SWIPP(SS):SR=SWIPR(SS)
2840 LAYER=LAY
2845 IF LAYER=1 THEN GOTO 2850 ELSE GOTO 2940
2850 T=AN1 : HT=HEIGHT
2860 EXAMINE=EXAMINE+1
2865 IF EXAMINE > 3 THEN GOTO 2870 ELSE GOTO 2875
2870 PRINT "EXAMINE > 3" : STOP
2875 MMM=1
2880 IF FIX(LP(LAYER,MMM)/L(BOX))*FIX(WP(LAYER,MMM)/W(BOX))>0
   OR FIX(LP(LAYER,MMM)/W(BOX))*FIX(WP(LAYER,MMM)/L(BOX))>0
   THEN GOTO 2900 ELSE GOTO 2920
2900 MINL=LAYER : MINA=MMM : GOTO 3040
2920 MMM=MMM+1 : IF MMM>T THEN LAYER=LAYER+1 : GOTO 2940

   GOTO 2940 ELSE GOTO 2880
2940 IF LAYER=2 THEN GOTO 2960 ELSE GOTO 2980
2960 MMM=1 : T=AN2 : GOTO 2880

2970 IF LAY=1 THEN GOTO 2971 ELSE HT=HEIGHT : GOTO 2975
2971   HT=HEIGHT+300

2975 GOTO 2880
2980 IF LAYER=3 THEN GOTO 3000 ELSE GOTO 3020
3000 MMM=1 : T=AN3
```

```
3005 IF LAY=1 THEN GOTO 3006 ELSE HT=HEIGHT+300 : GOTO 3010
3006    HT=HEIGHT+600
3010 GOTO 2240
3020 IF BOX=N THEN GOTO 3220 ELSE GOTO 2640
3040 IF MMM+1 > T GOTO 3190 ELSE GOTO 3060
3060 FOR I=MMM+1 TO T
3080    IF LP(LAYER,I)*WP(LAYER,I)<LP(MINL,MINA)*WP(MINL,MINA)
           THEN GOTO 3100 ELSE GOTO 3160
3100    IF FIX(LP(LAYER,I)/L(BOX))*FIX(WP(LAYER,I)/W(BOX))>0
           OR FIX(LP(LAYER,I)/W(BOX))*FIX(WP(LAYER,I)/L(BOX))>0
           THEN GOTO 3120 ELSE GCTO 3160
3120    MINL=LAYER
3140    MINA=I
3160 NEXT I
3170 Z=SS
3190 GOSUB 4000
3200 GOSUB 5000
3220 RETURN
4000 '
4020 '
4040 'SUBROUTINE ORIENTATION
4060 PRINT "Load type";BOX;"box onto layer";MINL;"and area";
           MINA;"of pallet"
4080 PRINT "STL=";STL(MINL,MINA);"STW=";STW(MINL,MINA);"FINL=";
           FINL(MINL,MINA);"FINW=";FINW(MINL,MINA)
4090 TTIME$=TIME$ : INPUT B$
4091 TIME$=TTIME$
4100 IF FIX(LP(MINL,MINA)/L(BOX))*FIX(WP(MINL,MINA)/W(BOX))>=
        FIX(LP(MINL,MINA)/W(BOX))*FIX(WP(MINL,MINA)/L(BOX))
        THEN GOTO 4108 ELSE GOTO 4115
4108 LENT=L(BOX) : WIDT=W(BOX) : SR=SR-900
4109 FX=-140+(100*(STW(MINL,MINA)-1))+(100*L(BOX)/2):FZ=HT+309
4110 FP=-883:FR=-843:FY=3986-(100*(STL(MINL,MINA)-1))-(100*W(BOX)/2)
4111 GOTO 4118
4115 LENT=W(BOX) : WIDT=L(BOX)
4116 FX=-140+(100*(STW(MINL,MINA)-1))+(100*L(BOX)/2):FZ=HT+309
```

```
4117 FP=-883:FR=-843:FY=3986-(100*(STL(MINL,MINA)-1))-(100*W(BOX)/2)
4118 IF WIP(Z)=BOX THEN GOTO 4119 ELSE GOTO 4121
4119 WIP(Z)=0:SWIPX(Z)=0:SWIPY(Z)=0:SWIPZ(Z)=0
4120 SWIPP(Z)=0:SWIPR(Z)=0:TRUE=0:GOTO 4124
4121 IF OBOX=BOX THEN GOTO 4124 ELSE GOTO 4125
4124 GOSUB 4500 : GOSUB 15000 : GOTO 4160
4125 SX=4307 : SY=-908 : SZ=1326 : SP=-908 : SR=1006
4126 IF LENT=L(BOX) AND WIDT=W(BOX) THEN SR=SR-900
4128 TRUE=1
4135 GOSUB 4500 : GOSUB 14000
4160 FOR I=STL(MINL,MINA) TO (STL(MINL,MINA)+LENT-1)
4180    FOR J=STW(MINL,MINA) TO (STW(MINL,MINA)+WIDT-1)
4200            GRID(MINL,I,J)=0
4210            COUNT=COUNT+1
4220            IF (MINL+1) > 3 THEN GOTO 4260 ELSE GOTO 4240
4240            GRID(MINL+1,I,J)=1
4260    NEXT J
4280 NEXT I
4290 EXAMINE=0
4300 BOXES(LEE)=BOX
4310 LEE=LEE+1
4330 RETURN
4500 IF STW(MINL,MINA)=1 THEN GOTO 4505 ELSE GOTO 4535
4505 IF STL(MINL,MINA)=1 THEN GOTO 4530 ELSE GOTO 4515
4515 IF BOX=2 THEN SR=SR+900 : FR=FR+900
4530 RETURN
4535 IF STL(MINL,MINA)=1 THEN GOTO 4540 ELSE GOTO 4545
4540 IF GRID(MINL,FINL(MINL,MINA)+1,STW(MINL,MINA))=0
        THEN GOTO 4545 ELSE GOTO 4530
4545 IF BOX=2 THEN SR=SR+900 : FR=FR+900
4560 GOTO 4530
5000 '
5015 '
5030 'SUBROUTINE LPART
5045 LAYER=1
5060 TAN1=1
```

```
5075 TAN2=1
5090 TAN3=1
5105 IF LAY=2 THEN T=TAN2 : LAYER=2 : GOTO 5150 ELSE GOTO 5120
5120 IF LAY=3 THEN T=TAN3 : LAYER=3 : GOTO 5150 ELSE GOTO 5135
5135 T=TAN1
5150 TSTL(LAYER,T)=1
5165 TSTW(LAYER,T)=1
5180 TFINL(LAYER,T)=LENP
5195 TFINW(LAYER,T)=WIDP
5210 FOR I=TSTW(LAYER,T) TO TFINW(LAYER,T)
5225    FOR J=TSTL(LAYER,T) TO TFINL(LAYER,T)
5240        IF GRID(LAYER,J,I)=0 THEN GOTO 5255 ELSE GOTO 5315
5255    NEXT J
5270 NEXT I
5285 T=T-1
5300 GOTO 6050
5315 FOR I=TSTW(LAYER,T) TO TFINW(LAYER,T)
5330    FOR J=TSTL(LAYER,T) TO TFINL(LAYER,T)
5345        IF GRID(LAYER,J,I)=1 THEN GOTO 5360 ELSE GOTO 6020
5360            TSTL(LAYER,T)=J
5375            TSTW(LAYER,T)=I
5390            TFINW(LAYER,T)=I
5405            FOR JJ=TSTL(LAYER,T) TO TFINL(LAYER,T)
5420                IF GRID(LAYER,JJ,I)=0 THEN GOTO 5435
                        ELSE GOTO 5450

5435                    TFINL(LAYER,T)=JJ-1 : GOTO 5885
5450            NEXT JJ
5465            IF TFINL(LAYER,T)=LENP AND TSTL(LAYER,T)<L(1)
5480            K=TFINL(LAYER,T)
5495            FOR KK=1 TO WIDP
5510                IF GRID(LAYER,K,KK)=1 THEN GOTO 5525
                       ELSE GOTO 5555

5525                TSTW(LAYER,T)=KK : TFINW(LAYER,T)=KK
5540                GOTO 5570
5555            NEXT KK
5570            FOR KK=TSTW(LAYER,T)+1 TO WIDP
```

```
5585                    IF GRID(LAYER,K,KK)=1 THEN GOTO 5600
                           ELSE GOTO 5630
5600                    TFINW(LAYER,T)=KK
5615                NEXT KK
5630                TWP(LAYER,T)=TFINW(LAYER,T)-TSTW(LAYER,T)+1
5645                FOR KK=TFINL(LAYER,T)-1 TO TSTL(LAYER,T) STEP -1
5660                    FOR KKK=TSTW(LAYER,T) TO TFINW(LAYER,T)
5675                           IF GRID(LAYER,KK,KKK)=1 THEN GOTO 5690
                                  ELSE GOTO 5735
5690                    NEXT KKK
5705                   TSTL(LAYER,T)=KK
5720                NEXT KK
5735                TLP(LAYER,T)=TFINL(LAYER,T)-TSTL(LAYER,T)+1
5875                GOTO 6245
5885                TLP(LAYER,T)=TFINL(LAYER,T)-TSTL(LAYER,T)+1
5900                FOR JJJJ=TSTL(LAYER,T) TO TFINL(LAYER,T)
5915                        IF TFINW(LAYER,T)+1>WIDP THEN GOTO 5916
                                ELSE GOTO 5930
5916                            TFINW(LAYER,T)=WIDP:GOTO 5990
5930                        IF GRID(LAYER,JJJJ,TFINW(LAYER,T)+1)=0
                                THEN GOTO 5990 ELSE GOTO 5945
5945                NEXT JJJJ
5960                TFINW(LAYER,T)=TFINW(LAYER,T)+1
5975                GOTO 5900
5990                TWP(LAYER,T)=TFINW(LAYER,T)-TSTW(LAYER,T)+1
6005                GOTO 6245
6020       NEXT J
6035 NEXT I
6050 IF LAYER=1 THEN TAN1=T
6065 IF LAYER=2 THEN TAN2=T
6075 IF LAYER=3 THEN TAN3=T
6080 IF LAYER+1 > 3 THEN GOTO 6118 ELSE GOTO 6081
6081 AVAIL=0
6082 FOR I=1 TO T
6083    IF TLP(LAYER,I)<L(N) OR TWP(LAYER,I)<W(N) THEN GOTO 6084
```

```
           ELSE AVAIL=1 : GOTO 6093
6084    IF TLP(LAYER,I)<W(N) OR TWP(LAYER,I)<L(N) THENGOTO 6085
           ELSE AVAIL=1 : GOTO 6093
6085    IF TFINL(LAYER,I)=LENP OR TFINW(LAYER,I)=WIDP
           THEN GOTO 6091 ELSE GOTO 6086
6086    FOR J=TSTL(LAYER,I) TO TFINL(LAYER,I)
6087            FOR K=TSTW(LAYER,I) TO TFINW(LAYER,I)
6088                    GRID(LAYER,J,K)=0 : GRID(LAYER+1,J,K)=1
6089            NEXT K
6090    NEXT J
6091 NEXT I
6093 IF AVAIL=0 THEN GOTO 6095 ELSE GOTO 6118
6095 FOR I=1 TO T
6097    IF TFINL(LAYER,I)=LENP OR TFINW(LAYER,I)=WIDP
           THEN GOTO 6099 ELSE GOTO 6109
6099    FOR J=TSTL(LAYER,I) TO TFINL(LAYER,I)
6101            FOR K=TSTW(LAYER,I) TO TFINW(LAYER,I)
6103                    GRID(LAYER,J,K)=0 : GRID(LAYER+1,J,K)=1
6105            NEXT K
6107    NEXT J
6109 NEXT I
6118 LAYER=LAYER+1
6119 IF LAYER=2 THEN GOTO 6125 ELSE GOTO 6155
6125 T=TAN2
6140 GOTO 5150
6155 IF LAYER=3 THEN GOTO 6170 ELSE GOTO 6200
6170 T=TAN3
6185 GOTO 5150
6200 GOSUB 8000
6215 GOSUB 10000
6230 RETURN
6245 T=T+1
6260 IF TSTL(LAYER,T-1)=1 THEN GOTO 6275 ELSE GOTO 6335
6275 IF TFINW(LAYER,T-1)=WIDP THEN GOTO 6455 ELSE GOTO 6290
6290 TSTL(LAYER,T)=1
```

```
6305 TSTW(LAYER,T)=TFINW(LAYER,T-1)+1
6320 GOTO 6485
6335 FOR J=1 TO WIDP
6350    FOR I=1 TO TSTL(LAYER,T-1)-1
6365          IF GRID(LAYER,I,J)=1 THEN GOTO 6380 ELSE GOTO 6425
6380          TSTL(LAYER,T)=I
6395          TSTW(LAYER,T)=J
6410          GOTO 6485
6425    NEXT I
6440 NEXT J
6455 T=T-1
6470 GOTO 6050
6485 IF  TSTW(LAYER,T)  >  TFINW(LAYER,T-1)  THEN  GOTO  6486  ELSE
       GOTO 6490
6486     TFINL(LAYER,T)=TFINL(LAYER,T-1) : GOTO 6500
6490     TFINL(LAYER,T)=TSTL(LAYER,T-1)-1
6500 TFINW(LAYER,T)=WIDP
6515 GOTO 5210
7000 '
7015 '
7030 'SUBROUTINE CHK-AREA
7045 IF LAY=1 THEN GOTO 7060 ELSE 7150
7060 T=AN1
7070 HEIGHT(1)=0
7075 FOR I=1 TO T
7090     IF LP(LAY,I)>=L(N) AND WP(LAY,I)>=W(N) THEN GOTO 7270
           ELSE GOTO 7105
7105     IF LP(LAY,I)>=W(N) AND WP(LAY,I)>=L(N) THEN GOTO 7270
           ELSE GOTO 7120
7120 NEXT I
7135 LAY=LAY+1
7150 IF LAY=2 THEN GOTO 7165 ELSE GOTO 7195
7165 T=AN2
7170 HEIGHT(2)=300
```

```
7180 GOTO 7075
7195 IF LAY=3 THEN GOTO 7210 ELSE GOTO 7240
7210 T=AN3
7215 HEIGHT(3)=600
7225 GOTO 7075
7240 GOSUB 20120
7255 END
7270 RETURN
7500 '
7515 '
7530 ' SUBROUTINE EXACT
7545 IF LAY=1 THEN T=AN1 : GOTO 7590 ELSE GOTO 7560
7560 IF LAY=2 THEN T=AN2 : GOTO 7590 ELSE GOTO 7575
7575 IF LAY=3 THEN T=AN3 : GOTO 7590
7590 FOR SIMON=1 TO T
7605     IF L(TEMP)<=LP(LAY,SIMON) AND W(TEMP)<=WP(LAY,SIMON)
            THEN GOTO 7635 ELSE GOTO 7620
7620     IF L(TEMP)<=WP(LAY,SIMON) AND W(TEMP)<=LP(LAY,SIMON)
            THEN GOTO 7635ELSE GOTO 7710
7635     IF TEMP-1 > 0 THEN GOTO 7650 ELSE GOTO 7680
7650     IF L(TEMP-1)>LP(LAY,SIMON) AND W(TEMP-1)>WP(LAY,SIMON)
            THEN GOTO 7680 ELSE GOTO 7665
7665     IF L(TEMP-1)>WP(LAY,SIMON) AND W(TEMP-1)>LP(LAY,SIMON)
            THEN GOTO 7680 ELSE GOTO 7710
7680     CHECK=1
7695     RETURN
7710 NEXT SIMON
7725 RETURN
8000 '
8020 '
8040 'SUBROUTINE WPART
8060 LAYER=1
8080 UAN1=1
8100 UAN2=1
8120 UAN3=1
8140 IF LAY=2 THEN T=UAN2 : LAYER=2 : GOTO 8200 ELSE GOTO 8160
```

```
8160 IF LAY=3 THEN T=UAN3 : LAYER=3 : GOTO 8200 ELSE GOTO 8180
8180 T=UAN1
8200 USTL(LAYER,T)=1
8220 USTW(LAYER,T)=1
8240 UFINL(LAYER,T)=LENP
8260 UFINW(LAYER,T)=WIDP
8280 FOR I=USTL(LAYER,T) TO UFINL(LAYER,T)
8300    FOR J=USTW(LAYER,T) TO UFINW(LAYER,T)
8320        IF GRID(LAYER,I,J)=0 THEN GOTO 8340 ELSE GOTO 8420
8340    NEXT J
8360 NEXT I
8380 T=T-1
8400 GOTO 8840
8420 FOR I=USTL(LAYER,T) TO UFINL(LAYER,T)
8440    FOR J=USTW(LAYER,T) TO UFINW(LAYER,T)
8460        IF GRID(LAYER,I,J)=1 THEN GOTO 8480 ELSE GOTO 8800
8480        USTW(LAYER,T)=J
8500        USTL(LAYER,T)=I
8520        UFINL(LAYER,T)=I
8540        FOR JJ=USTW(LAYER,T) TO UFINW(LAYER,T)
8560            IF GRID(LAYER,I,JJ)=0 THEN GOTO 8580 ELSE GOTO 8600
8580            UFINW(LAYER,T)=JJ-1 :GOTO 8620
8600        NEXT JJ
8620        UWP(LAYER,T)=UFINW(LAYER,T)-USTW(LAYER,T)+1
8640        FOR JJJJ=USTW(LAYER,T) TO UFINW(LAYER,T)
8660            IF UFINL(LAYER,T)+1 >LENP THEN GOTO 8661
                   ELSE GOTO 8680
8661               UFINL(LAYER,T)=LENP : GOTO 8760
8680            IF GRID(LAYER,UFINL(LAYER,T)+1,JJJJ)=0
                   THEN GOTO 8760 ELSE GOTO 8700
8700        NEXT JJJJ
8720        UFINL(LAYER,T)=UFINL(LAYER,T)+1
8740        GOTO 8640
8760        ULP(LAYER,T)=UFINL(LAYER,T)-USTL(LAYER,T)+1
8780        GOTO 9060
```

```
8800    NEXT J
8820 NEXT I
8840 IF LAYER=1 THEN UAN1=T
8860 IF LAYER=2 THEN UAN2=T
8880 IF LAYER=3 THEN UAN3=T
8900 LAYER=LAYER+1
8920 IF LAYER=2 THEN GOTO 8940 ELSE GOTO 8980
8940 T=UAN2
8960 GOTO 8200
8980 IF LAYER=3 THEN GOTO 9000 ELSE GOTO 9059
9000 T=UAN3
9020 GOTO 8200
9059 RETURN
9060 T=T+1
9080 IF USTW(LAYER,T-1)=1 THEN GOTO 9100 ELSE GOTO 9220
9100 IF UFINL(LAYER,T-1)=LENP AND UFINW(LAYER,T-1)=WIDP
9101    THEN GOTO 9400 ELSE GOTO 9120
9120 IF USTL(LAYER,T-1)=1 AND UFINL(LAYER,T-1)=LENP
        THEN GOTO 9140 ELSE GOTO 9200
9140 USTL(LAYER,T)=1 : USTW(LAYER,T)=UFINW(LAYER,T-1)+1
9160 UFINL(LAYER,T)=LENP : UFINW(LAYER,T)=WIDP
9180 GOTO 8280
9200 USTL(LAYER,T)=UFINL(LAYER,T-1)+1 : USTW(LAYER,T)=1 :
     GOTO 9440
9220 IF T-1=1 THEN GOTO 9240 ELSE GOTO 9400
9240 FOR J=1 TO LENP
9260    FOR I=1 TO USTW(LAYER,T-1)-1
9280        IF GRID(LAYER,J,I)=1 THEN GOTO 9300 ELSE GOTO 9360
9300        USTL(LAYER,T)=J
9320        USTW(LAYER,T)=I
9340        GOTO 9440
9360    NEXT I
9380 NEXT J
9400 T=T-1
9420 GOTO 8840
9440 UFINL(LAYER,T)=LENP
```

```
9460 IF USTL(LAYER,T) > UFINL(LAYER,T-1) THEN UFINW(LAYER,T)=WIDP
          ELSE UFINW(LAYER,T)=USTW(LAYER,T-1)-1
9480 GOTO 8280
10000 '
10015 '
10030 'SUBROUTINE SUB-AREA
10045 LAYER=1
10060 IF LAY=2 THEN GOTO 10195 ELSE GOTO 10075
10075 IF LAY=3 THEN GOTO 10195 ELSE GOTO 10090
10090 T=TAN1
10105 MAX1(LAYER)=TLP(LAYER,1)*TWP(LAYER,1)
10120 FOR I=1 TO T
10135    IF TLP(LAYER,I)*TWP(LAYER,I)>=MAX1(LAYER)
            THEN GOTO 10150 ELSE GOTO 10180
10150    MAX1(LAYER)=TLP(LAYER,I)*TWP(LAYER,I)
10165    HIGH(LAYER)=I
10180 NEXT I
10195 LAYER=LAYER+1
10210 IF LAYER=2 THEN GOTO 10225 ELSE GOTO 10255
10225 T=TAN2
10240 GOTO 10105
10255 IF LAYER=3 THEN GOTO 10270 ELSE GOTO 10300
10270 T=TAN3
10275 IF T=0 THEN GOTO 10300 ELSE GOTO 10285
10285 GOTO 10105
10300 LAYER=1
10315 IF LAY=2 THEN GOTO 10435 ELSE GOTO 10330
10330 IF LAY=3 THEN GOTO 10435 ELSE GOTO 10345
10345 T=UAN1
10360 MAX2(LAYER)=ULP(LAYER,1)*UWP(LAYER,1)
10375 FOR I=1 TO T
10390    IF ULP(LAYER,I)*UWP(LAYER,I)>=MAX2(LAYER)
            THEN GOTO 10405 ELSE GOTO 10420
10405    MAX2(LAYER)=ULP(LAYER,I)*UWP(LAYER,I)
10420 NEXT I
10435 LAYER=LAYER+1
```

```
10450 IF LAYER=2 THEN GOTO 10465 ELSE GOTO 10495
10465 T=UAN2
10480 GOTO 10360
10495 IF LAYER=3 THEN GOTO 10510 ELSE GOTO 10540
10510 T=UAN3
10515 IF T=0 THEN GOTO 10540 ELSE GOTO 10525
10525 GOTO 10360
10540 LAYER=1
10555 IF LAY=2 THEN GOTO 10900 ELSE GOTO 10570
10570 IF LAY=3 THEN GOTO 10900 ELSE GOTO 10585
10585 IF MAX1(LAYER)>MAX2(LAYER) THEN
10600 IF MAX2(LAYER) = MAX1(LAYER) AND MAX1(LAYER)=L(1)*W(1)
         THEN GOTO 10615 ELSE GOTO 10945
10615 IF TSTL(LAYER,HIGH(LAYER))>1 THEN GOTO 10630 ELSE GOTO 10945
10630 IF LAYER=1 THEN GOTO 10645 ELSE GOTO 10690
10645 T=TAN1
10660 AN1=TAN1
10675 GOTO 10780
10690 IF LAYER=2 THEN GOTO 10705 ELSE GOTO 10750
10705 T=TAN2
10720 AN2=TAN2
10735 GOTO 10780
10750 T=TAN3
10765 AN3=TAN3
10780 FOR I=1 TO T
10795     STL(LAYER,I)=TSTL(LAYER,I)
10810     STW(LAYER,I)=TSTW(LAYER,I)
10825     FINL(LAYER,I)=TFINL(LAYER,I)
10840     FINW(LAYER,I)=TFINW(LAYER,I)
10855     WP(LAYER,I)=TWP(LAYER,I)
10870     LP(LAYER,I)=TLP(LAYER,I)
10885 NEXT I
10900 LAYER=LAYER+1
10915 IF LAYER=2 THEN GOTO 10585 ELSE GOTO 10930
10930 IF LAYER=3 AND T=0 THEN GOTO 11305 ELSE GOTO 10585
10945 IF LAYER=1 THEN GOTO 10960 ELSE GOTO 11005
```

```
10960 T=UAN1
10975 AN1=UAN1
10990 GOTO 11095
11005 IF LAYER=2 THEN GOTO 11020 ELSE 11065
11020 T=UAN2
11035 AN2=UAN2
11050 GOTO 11095
11065 T=UAN3
11080 AN3=UAN3
11095 FOR I=1 TO T
11110    STL(LAYER,I)=USTL(LAYER,I)
11125    STW(LAYER,I)=USTW(LAYER,I)
11140    FINL(LAYER,I)=UFINL(LAYER,I)
11155    FINW(LAYER,I)=UFINW(LAYER,I)
11170    WP(LAYER,I)=UWP(LAYER,I)
11185    LP(LAYER,I)=ULP(LAYER,I)
11200 NEXT I
11215 LAYER=LAYER+1
11230 IF LAYER=2 THEN GOTO 10585 ELSE GOTO 11245
11245 IF LAYER=3 THEN GOTO 10585 ELSE GOTO 11260
11260 IF LAY=3 THEN GOTO 11275 ELSE GOTO 11305
11275 LAYER=3 : T=AN3
11290 GOTO 11365
11305 IF LAY=2 THEN GOTO 11320 ELSE GOTO 11350
11320 LAYER=2 : T=AN2
11335 GOTO 11365
11350 LAYER=1 : T=AN1
11365 FOR I=1 TO T
11380   PRINT "STL(";I;") OF LAYER";LAYER;"=";STL(LAYER,I);"     ";
        "STW(";I;") OF LAYER";LAYER;"=";STW(LAYER,I)
11395   PRINT "FINL(";I;") OF LAYER";LAYER;"=";FINL(LAYER,I);"    ";
        "FINW("'O;") OF LAYER";LAYER;"=";FINW(LAYER,I)
11410   PRINT "LP(";I;") OF LAYER";LAYER;"=";LP(LAYER,I);"      ";
        "WP(";I;") OF LAYER";LAYER;"=";WP(LAYER,I)
11425 NEXT I
11440 LAYER=LAYER+1
```

```
11455 IF LAYER=2 THEN T=AN2 : GOTO 11365 ELSE GOTO 11470
11470 IF LAYER=3 THEN T=AN3 : GOTO 11365 ELSE GOTO 11490
11490 TTIME$=TIME$ : INPUT B$
11491 TIME$=TTIME$ : RETURN
12000 '
12010 '
12020 'VISION SUBROUTINE
12030 DEF SEG=&H8000 : 'TAKES A
12040 POKE &H8000,0 :  'PICTURE
12050 FOR DELAY=1 TO 50:NEXT : 'JUST A DELAY
12060 DEF SEG=&H3000 : 'calls the machine language subroutine
12070 CALL ABSOLUTE (ADDRESS) : 'TO DISPLAY THE PICTURE
12080 DEF SEG=&HA000
12090 LOCATE 14,64:PRINT "PROCESSING...."
12100 REM CLS
12120 FOR TT=8330 TO 60287!
12125    IF TT > 9610 THEN GOTO 13432 ELSE GOTO 12130
12130    L=PEEK(TT)
12140    PIXEL=INT(L/16)
12160    IF PIXEL<8 THEN GOTO 12170 ELSE GOTO 12380
12170    HON1=TT
12180    FOR K=(HON1+1) TO (HON1+4)
12190            L=PEEK(K)
12200            PIXEL=INT(L/16)
12220            IF PIXEL < 8 THEN GOTO 12230 ELSE GOTO 12380
12230   NEXT K
12240   FOR K=(HON1+5 ) TO 60287!
12250           L=PEEK(K)
12260           PIXEL=INT(L/16)
12270           IF PIXEL > 8 THEN GOTO 12280 ELSE 12350
12280           EHON=K
12290           FOR KK=(EHON+1) TO (EHON+4)
12300               L=PEEK(KK)
12310               PIXEL=INT(L/16)
12320               IF PIXEL > 8 THEN GOTO 12330 ELSE GOTO 12350
12330           NEXT KK
```

```
12340            GOTO 12360
12350    NEXT K
12360    MID=INT((EHON+HON1)/2+.5)
12370    GOTO 12390
12380 NEXT TT
12390 FOR TT=9610 TO 60287!
12400    L=PEEK(TT)
12410    PIXEL=INT(L/16)
12430    IF PIXEL < 8 THEN GOTO 12440 ELSE GOTO 12520
12440    HON2=TT
12450    FOR K=(HON2+1) TO (HON2+4)
12460        L=PEEK(K)
12470        PIXEL=INT(L/16)
12490        IF PIXEL < 8 THEN GOTO 12500 ELSE GOTO 12520
12500    NEXT K
12510    GOTO 12530
12520 NEXT TT
12530 XHON1=((HON1/128)-FIX(HON1/128))*128
12540 XHON2=((HON2/128)-FIX(HON2/128))*128
12550 YHON1=FIX(HON1/128)
12560 YHON2=FIX(HON2/128)
12570 IF (XHON2-XHON1)=0 THEN NSR=0 : STEPS=0 :

          GOTO 12610 ELSE GOTO 12580

12580 STEPS=XHON2-XHON1 : BASES=STEPS*2.5725
12590 ANG=(ATN((YHON2-YHON1)/BASES)*180)/3.141592654#
12600 IF ANG > 0 THEN NSR=10*(90-ANG) ELSE NSR=-10*(90+ANG)
12610 TM=ABS(TAN(ANG*3.141592654#/180))
12620 FOR J=1 TO 300
12630   IF STEPS < 0 THEN TT=MID-128*J+INT(J/TM)
12640   IF STEPS = 0 THEN TT=MID-128*J
12650   IF STEPS > 0 THEN TT=MID-128*J-INT(J/TM)
12660   L=PEEK(TT)
12670   PIXEL=INT(L/16)
12690   IF PIXEL< 8 THEN GOTO 12810 ELSE GOTO 12700
12700   TOPHON=TT
12710   FOR I=J TO J+4
```

```
12720         IF STEPS < 0 THEN TT=MID-128*I+INT(I/TM)
12730         IF STEPS = 0 THEN TT=MID-128*I
12740         IF STEPS > 0 THEN TT=MID-128*I-INT(I/TM)
12750         L=PEEK(TT)
12760         PIXEL=INT(L/16)
12780   IF PIXEL > 8 THEN GOTO 12790 ELSE GOTO 12810
12790   NEXT I
12800   GOTO 12820
12810 NEXT J
12820 FOR J=1 TO 300
12830   IF STEPS < 0 THEN TT=MID+128*J-INT(J/TM)
12840   IF STEPS = 0 THEN TT=MID+128*J
12850   IF STEPS > 0 THEN TT=MID+128*J+INT(J/TM)
12860   L=PEEK(TT)
12870   PIXEL=INT(L/16)
12890   IF PIXEL < 8 THEN GOTO 13020 ELSE GOTO 12900
12900   BOTHON=TT
12910   FOR I=J TO J+4
12920   IF STEPS < 0 THEN GOTO 12930 ELSE GOTO 12950
12930           IF STEPS < 0 THEN TT=MID+128*I-INT(I/TM)
12940           IF STEPS = 0 THEN TT=MID+128*I
12950           IF STEPS > 0 THEN TT=MID+128*I+INT(I/TM)
12960           L=PEEK(TT)
12970           PIXEL=INT(L/16)
12990           IF PIXEL > 8 THEN GOTO 13000 ELSE GOTO 13020
13000   NEXT I
13010   GOTO 13030
13020 NEXT J
13030 TTM=ABS(TAN((90-ANG)*3.141592654#/180))
13040 FOR J=1 TO 300
13050   IF STEPS < 0 THEN TT=HON1+128*INT(J*TTM)+J
13060   IF STEPS = 0 THEN TT=HON1+J
13070   IF STEPS > 0 THEN TT=HON1-128*INT(J*TTM)+J
13080   L=PEEK(TT)
13090   PIXEL=INT(L/16)
13110   IF PIXEL < 8 THEN GOTO 13230 ELSE GOTO 13120
```

```
13120   TOPVER=TT
13130   FOR I=J TO J+4
13140           IF STEPS < 0 THEN TT=HON1+128*INT(I*TTM)+I
13150           IF STEPS = 0 THEN TT=HON1+I
13160           IF STEPS > 0 THEN TT=HON1-128*INT(I*TTM)+I
13170           L=PEEK(TT)
13180           PIXEL=INT(L/16)
13200           IF PIXEL > 8 THEN GOTO 13210 ELSE GOTO 13230
13210   NEXT I
13220   GOTO 13240
13230 NEXT J
13240 TOPL=FIX(TOPHON/128)
13250 BOTL=FIX(BOTHON/128)
13260 LENGTH=BOTL-TOPL-1
13270 IF ANG<>0 THEN LENGTH=LENGTH/SIN(ABS(ANG*3.141592654#/180))
13280 RIGW=((TOPVER/128)-FIX(TOPVER/128))*128
13290 LEFW=XHON1
13300 WID=RIGW-LEFW
13310 IF ANG<>0 THEN WID=WID*SIN(ABS(ANG*3.141592654#/180))
13320 LENGTH=LENGTH*.0706333333# : WID=WID*.1756333333#
13322 AREA=LENGTH*WID
13330 E$="THE ANGLE = " : PRINT E$,ANG;"DEGREE"
13335 G$="OFF CENTER IN Y AXIS":PRINT G$,(CENY-CENTERY);"PIXEL"
13340 E$="LENGTH OF THE BOX = ":PRINT E$,LENGTH;"CM"
13350 F$="WIDTH OF THE BOX  = ":PRINT F$,WID;"CM"
13360 F$="AREA OF THE BOX   =":PRINT F$,AREA;"SQ. CM"
13362 NSY=INT((CENY-CENTERY)*.0706333333#*10)
13370 FOR I=1 TO N
13380   IF LENGTH>=(L(I)-0.4) AND LENGTH<=(L(I)+0.4)
           THEN GOTO 13390 ELSE GOTO 13410
13390   IF WID>=(W(I)-0.4) AND WID<=(W(I)+0.4)
           THEN GOTO 13400 ELSE GOTO 13410
13400   NBOX=I
13401 PRINT "IT IS A TYPE";NBOX;"BOX"
13402 GOTO 13440
13410 NEXT I
```

```
13420 PRINT "BOX DOES NOT BELONG TO GROUP"
13430 PRINT "PLEASE REMOVE BOX "
13431 GOTO 13450
13432 NBOX=99
13433 PRINT "There is no box in the measuring area !"
13440 TTIME$=TIME$ : INPUT B$
13442 TIME$=TTIME$ : CLS
13450 RETURN
14000 '
14020 '
14030 'CONVEYOR SUBROUTINE
14040 CLS
14060 A2=FIX(INP(784)/2^(SW2-9)) AND 1
14065 IF A2=1 THEN GOTO 14060 ELSE GOTO 14240
14240 S=127
14260 OUT 784-MOT1,127+S
14320 A1=FIX(INP(784)/2^(SW1-9)) AND 1
14340 A2=FIX(INP(784)/2^(SW2-9)) AND 1
14380 IF A1=0 AND A2=0 THEN GOTO 14320 ELSE GOTO 14500
14500 S=0
14520 OUT 784-MOT1,127+S
14522 IF TRUE=1 THEN GOTO 14524 ELSE GOTO 14540
14524 GOSUB 15000
14526 TRUE=0
14540 GOSUB 12000
14570 RETURN
15000 '
15020 '
15040 'ROBOT OR TRANSMISSION SUBROUTINE
15060 GOSUB 16480
15080 GOSUB 16580
15100 PRINT #1,"SETPVC PIUP X";STR$(SX); :GOSUB 16580
15120 FOR I=1 TO 200
15140 NEXT I
15155 DY=SY+CY
15160 PRINT #1,"SETPVC PIUP Y";STR$(DY); :GOSUB 16580
```

```
15180 FOR I=1 TO 200
15200 NEXT I
15210 DZ=SZ+400
15220 PRINT #1,"SETPVC PIUP Z";STR$(DZ); :GOSUB 16580
15240 FOR I=1 TO 200
15260 NEXT I
15280 PRINT #1,"SETPVC PIUP P";STR$(SP); :GOSUB 16580
15300 FOR I=1 TO 200
15320 NEXT I
15335 DR=SR+CR
15340 PRINT #1,"SETPVC PIUP R";STR$(DR); :GOSUB 16580
15360 FOR I=1 TO 200
15380 NEXT I
15400 PRINT #1,"SETPVC PICKU X";STR$(SX); :GOSUB 16580
15420 FOR I=1 TO 200
15440 NEXT I
15455 DY=SY+CY
15460 PRINT #1,"SETPVC PICKU Y";STR$(DY); :GOSUB 16580
15480 FOR I=1 TO 200
15500 NEXT I
15540 PRINT #1,"SETPVC PICKU Z";STR$(SZ); :GOSUB 16580
15560 FOR I=1 TO 200
15580 NEXT I
15600 PRINT #1,"SETPVC PICKU P";STR$(SP); :GOSUB 16580
15620 FOR I=1 TO 200
15640 NEXT I
15655 DR=SR+CR
15660 PRINT #1,"SETPVC PICKU R";STR$(DR); :GOSUB 16580
15680 FOR I=1 TO 200
15700 NEXT I
15720 PRINT #1,"SETPVC UNLD X";STR$(FX); :GOSUB 16580
15740 FOR I=1 TO 200
15760 NEXT I
15780 PRINT #1,"SETPVC UNLD Y";STR$(FY); :GOSUB 16580
15800 FOR I=1 TO 200
15820 NEXT I
```

```
15830 DZ=FZ+400
15840 PRINT #1,"SETPVC UNLD Z";STR$(DZ); :GOSUB 16580
15860 FOR I=1 TO 200
15880 NEXT I
15900 PRINT #1,"SETPVC UNLD P";STR$(FP); :GOSUB 16580
15920 FOR I=1 TO 200
15940 NEXT I
15960 PRINT #1,"SETPVC UNLD R";STR$(FR); :GOSUB 16580
15980 FOR I=1 TO 200
16000 NEXT I
16020 PRINT #1,"SETPVC ULOAD X";STR$(FX); :GOSUB 16580
16040 FOR I=1 TO 200
16060 NEXT I
16080 PRINT #1,"SETPVC ULOAD Y";STR$(FY); :GOSUB 16580
16100 FOR I=1 TO 200
16120 NEXT I
16160 PRINT #1,"SETPVC ULOAD Z";STR$(FZ); :GOSUB 16580
16180 FOR I=1 TO 200
16200 NEXT I
16220 PRINT #1,"SETPVC ULOAD P";STR$(FP); :GOSUB 16580
16221 FOR I=1 TO 200
16222 NEXT I
16230 PRINT #1,"SETPVC ULOAD R";STR$(FR); :GOSUB 16580
16231 FOR I=1 TO 200
16232 NEXT I
16420 PRINT #1,"RUN ROBOT"
16440 CLOSE
16460 RETURN
16480 REM OPEN COMMUNICATION PORT
16500 OPEN "COM1:9600,N,8,1" AS #1
16520 GOSUB 16580
16540 PRINT #1,"noecho"
16560 RETURN
16580 REM WAITING FOR ACL PROMPT
16600 WHILE NOT EOF(1) : A$=INPUT$(1,#1) : PRINT A$; : WEND
16620 FOR I=1 TO 600
```

```
16640 NEXT I
16660 PRINT #1,""
16680 TIMEOUT% = 0
16720 WHILE EOF(1)
16780 WEND
16800 A$ = INPUT$(1,#1)
16820 IF A$ = ">" GOTO 16920
16840 TIMEOUT% = TIMEOUT% + 1
16860 IF TIMEOUT% < 300 GOTO 16700
16880 PRINT "WAITING FOR ACL RESPONSE"
16900 GOTO 16660
16920 CLS : RETURN
20000 '
20010 '
20020 'FINAL OUTPUT SUBROUTINE
20100 TTIME$=TIME$
20120 PERCENT=((LENP*WIDP)-COUNT)*100/(LENP*WIDP)
20140 PRINT "BOXES ON PALLET ARE AS FOLLOWS :"
20160 LEE = LEE - 1  :  EDRIC = EDRIC - 1
20180 FOR I=1 TO LEE
20200   PRINT I;":";BOXES(I)
20220 NEXT I
20240 IF EDRIC > 0 THEN GOTO 20260 ELSE GOTO 20320
20260 PRINT "MAXIMUM WORK IN PROCESS IS AS FOLLOWS :"
20280 FOR I=1 TO EDRIC
20300   PRINT I;":";MAXWIPS(I)
20320 NEXT I
20340 PRINT "THE PALLET IS FULL AND THE UTILIZATION IS ";
            PERCENT;"PERCENT"
20360 PRINT "THE TOTAL TIME IS";TTIME$
20380 INPUT A$
20400 RETURN
```

APPENDIX B

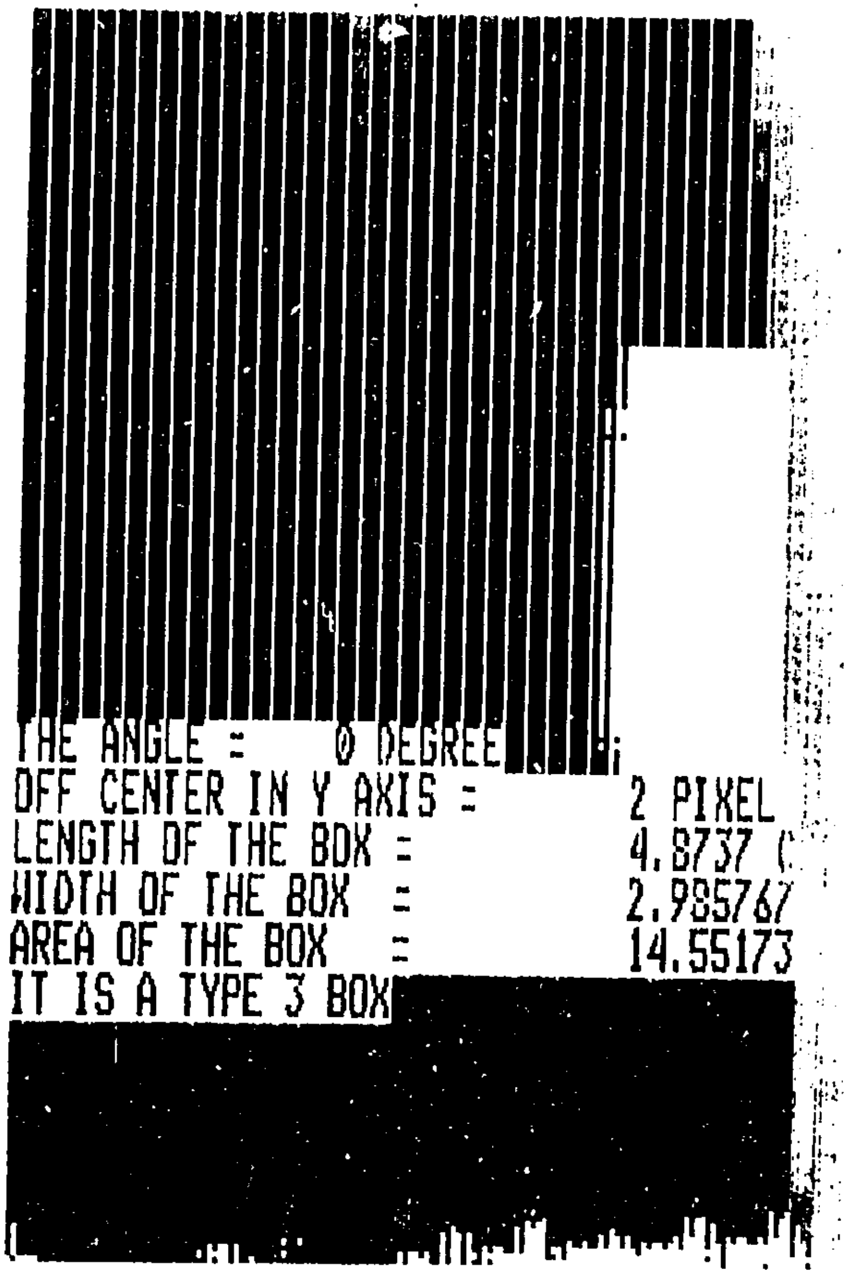
LISTING OF ACL ROBOT PROGRAM

"ROBOT"

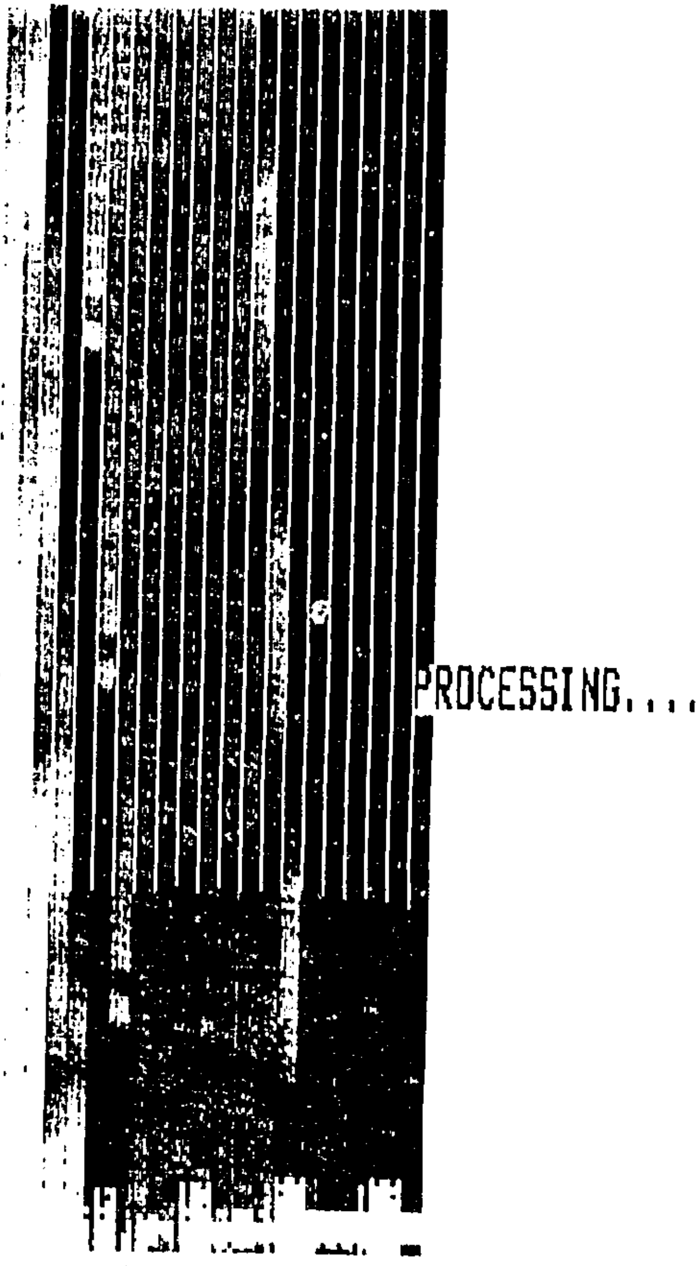
```
                    PROGRAM   ROBOT
                  *********************
SPEED       70
OPEN        1500
MOVEC       PIUP DUM1
DELAY       450
MOVEL       PICKU
DELAY       50
CLOSE       1800
DELAY       200
MOVEL       PIUP
DELAY       50
MOVEC       UNLD DUM2
DELAY       600
MOVEL       ULOAD
DELAY       50
OPEN        1500
MOVEL       UNLD
DELAY       50
MOVEC       HM DUM2
DELAY       300
END
```

APPENDIX C

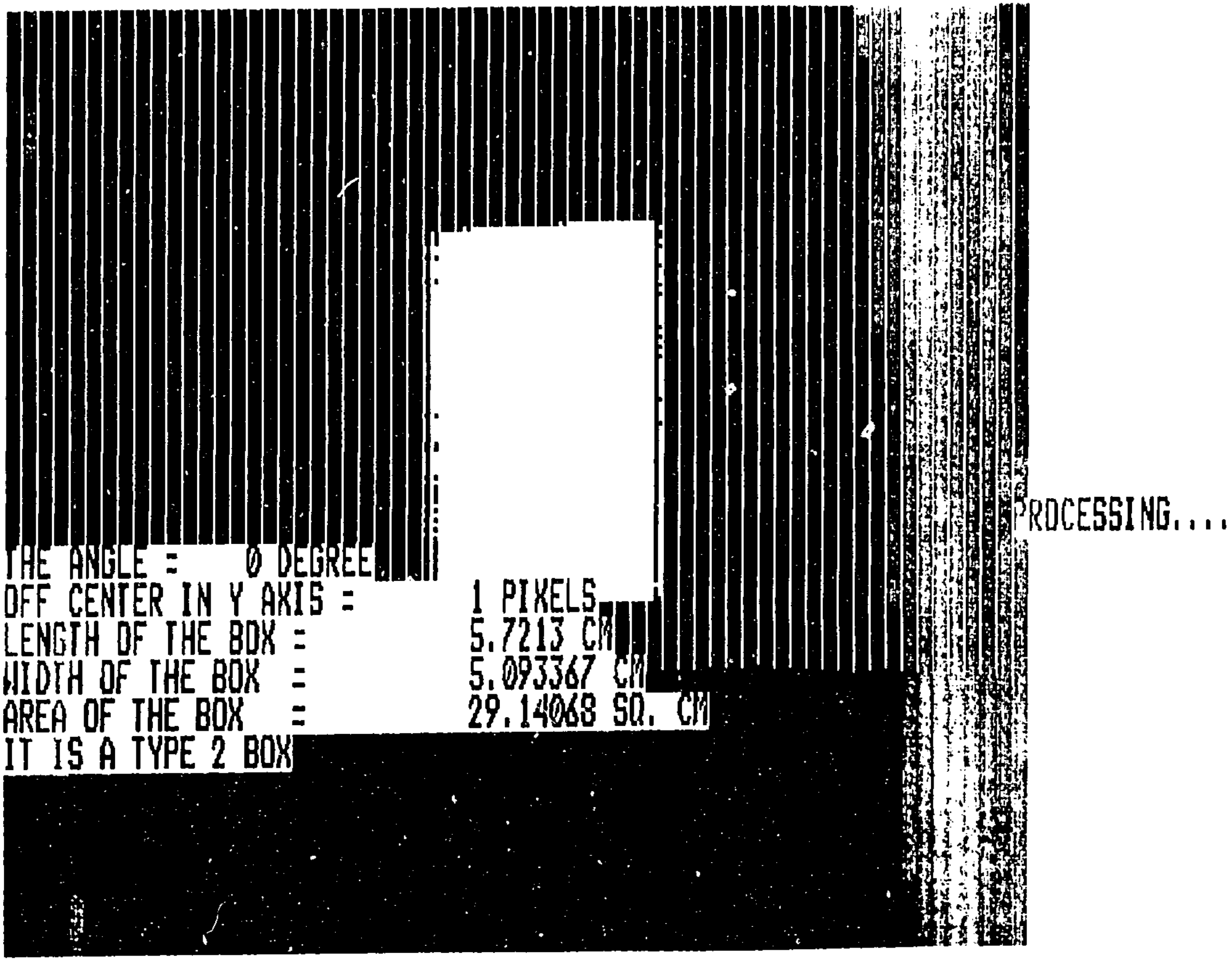
IMPLEMENTATION OUTPUT



```
THE ANGLE =    0 DEGREE
OFF CENTER IN Y AXIS =        2 PIXEL
LENGTH OF THE BOX =           4.8737 (
WIDTH OF THE BOX  =           2.985767
AREA OF THE BOX   =           14.55173
IT IS A TYPE 3 BOX
```



```
PROCESSING....
```



```
THE ANGLE =     0 DEGREE
OFF CENTER IN Y AXIS =          1 PIXELS
LENGTH OF THE BOX =             5.7213 CM
WIDTH OF THE BOX  =             5.093367 CM
AREA OF THE BOX   =             29.14068 SQ. CM
IT IS A TYPE 2 BOX
```

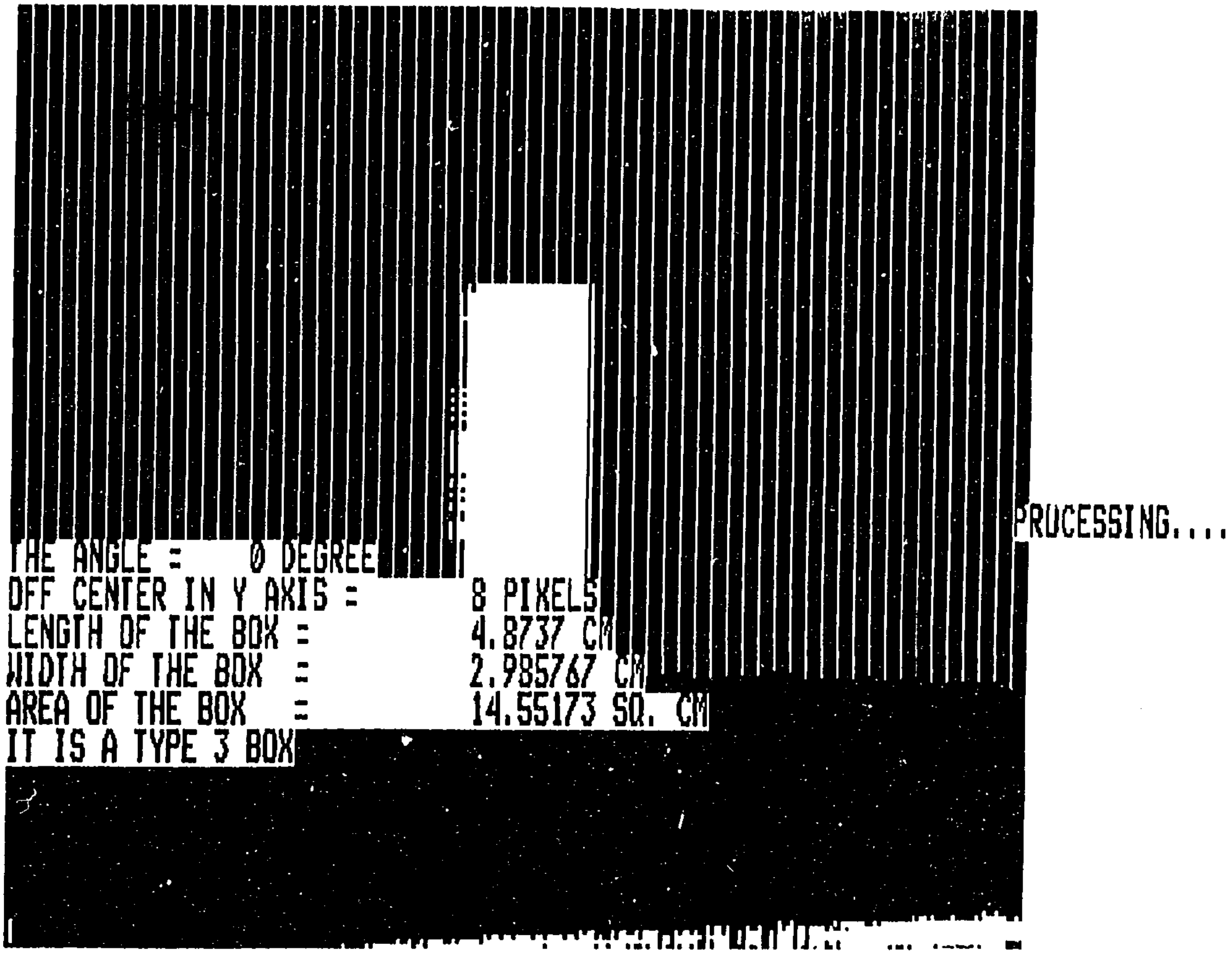
```
PROCESSING....
```

```
WIP( 1 )= 1
?
Load type 2 box onto layer 1 and area 1 of pallet
STL= 1 STW= 1 FINL= 12 FINW= 10
?
```

```
[illegible]
STL( 1 ) OF LAYER 1 = [illegible]      STW( 1 ) OF LAYER [illegible]
FINL( 1 ) OF LAYER 1 = 11      FINW( 1 ) OF LAYER 1 = [illegible]
LP( 1 ) OF LAYER 1 = 6         WP( 1 ) OF LAYER 1 = 10
STL( 2 ) OF LAYER 1 = 1        STW( 2 ) OF LAYER 1 = 6
FINL( 2 ) OF LAYER 1 = 6       FINW( 2 ) OF LAYER 1 = 10
LP( 2 ) OF LAYER 1 = 6         WP( 2 ) OF LAYER 1 = 5
STL( 1 ) OF LAYER 2 = 1        STW( 1 ) OF LAYER 2 = 1
FINL( 1 ) OF LAYER 2 = 6       FINW( 1 ) OF LAYER 2 = 5
LP( 1 ) OF LAYER 2 = 6         WP( 1 ) OF LAYER 2 = 5
?

Load type 2 box onto layer 1 and area 2 of pallet
STL= 1 STW= 6 FINL= 6 FINW= 10
?
```

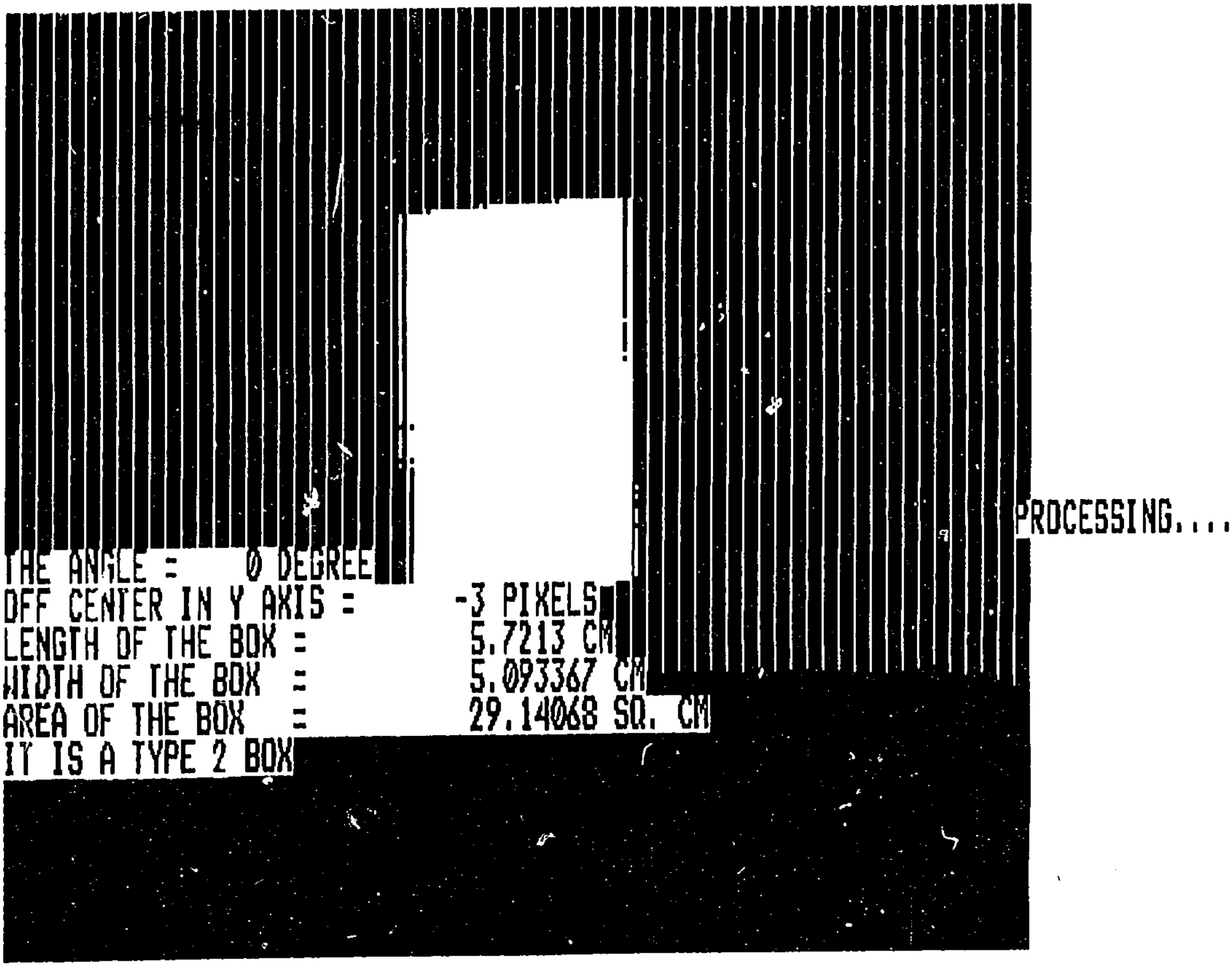
```
? 320
STL( 1 ) OF LAYER 1 = 7      STW( 1 ) OF LAYER 1 = 1
FINL( 1 ) OF LAYER 1 = 12     FINW( 1 ) OF LAYER 1 = 10
LP( 1 ) OF LAYER 1 = 6       WP( 1 ) OF LAYER 1 = 10
STL( 1 ) OF LAYER 2 = 1      STW( 1 ) OF LAYER 2 = 1
FINL( 1 ) OF LAYER 2 = 6     FINW( 1 ) OF LAYER 2 = 10
LP( 1 ) OF LAYER 2 = 6       WP( 1 ) OF LAYER 2 = 10
?
Load type 2 box onto layer 1 and area 1 of pallet
STL= 7 STW= 1 FINL= 12 FINW= 10
?
```



```
THE ANGLE =     0 DEGREE
OFF CENTER IN Y AXIS =        8 PIXELS
LENGTH OF THE BOX =           4.8737 CM
WIDTH OF THE BOX  =           2.985767 CM
AREA OF THE BOX   =           14.55173 SQ. CM
IT IS A TYPE 3 BOX
PROCESSING....
```

```
1#563
STL( 1 ) OF LAYER 1 = 7         STW( 1 ) OF LAYER 1 = 1
FINL( 1 ) OF LAYER 1 = 12        FINW( 1 ) OF LAYER 1 = 10
LP( 1 ) OF LAYER 1 = 6          WP( 1 ) OF LAYER 1 = 5
STL( 1 ) OF LAYER 2 = 1         STW( 1 ) OF LAYER 2 = 1
FINL( 1 ) OF LAYER 2 = 6        FINW( 1 ) OF LAYER 2 = 10
LP( 1 ) OF LAYER 2 = 6          WP( 1 ) OF LAYER 2 = 10
STL( 2 ) OF LAYER 2 = 7         STW( 2 ) OF LAYER 2 = 1
FINL( 2 ) OF LAYER 2 = 12        FINW( 2 ) OF LAYER 2 = 5
LP( 2 ) OF LAYER 2 = 6          WP( 2 ) OF LAYER 2 = 5
?
Load type 3 box onto layer 1 and area 1 of pallet
STL= 7 STW= 6 FINL= 12 FINW= 10
?
```

```
STL( 1 ) OF LAYER 1 = 10      STW( 1 ) OF LAYER 1 = 6
FINL( 1 ) OF LAYER 1 = 12      FINW( 1 ) OF LAYER 1 = 10
LP( 1 ) OF LAYER 1 = 3        WP( 1 ) OF LAYER 1 = 5
STL( 1 ) OF LAYER 2 = 1       STW( 1 ) OF LAYER 2 = 1
FINL( 1 ) OF LAYER 2 = 9      FINW( 1 ) OF LAYER 2 = 10
LP( 1 ) OF LAYER 2 = 9        WP( 1 ) OF LAYER 2 = 10
STL( 2 ) OF LAYER 2 = 10       STW( 2 ) OF LAYER 2 = 1
FINL( 2 ) OF LAYER 2 = 12      FINW( 2 ) OF LAYER 2 = 5
LP( 2 ) OF LAYER 2 = 3        WP( 2 ) OF LAYER 2 = 5
?
```



```
THE ANGLE =     0 DEGREE
OFF CENTER IN Y AXIS =       -3 PIXELS
LENGTH OF THE BOX =           5.7213 CM
WIDTH OF THE BOX  =           5.093367 CM
AREA OF THE BOX   =           29.14068 SQ. CM
IT IS A TYPE 2 BOX
```

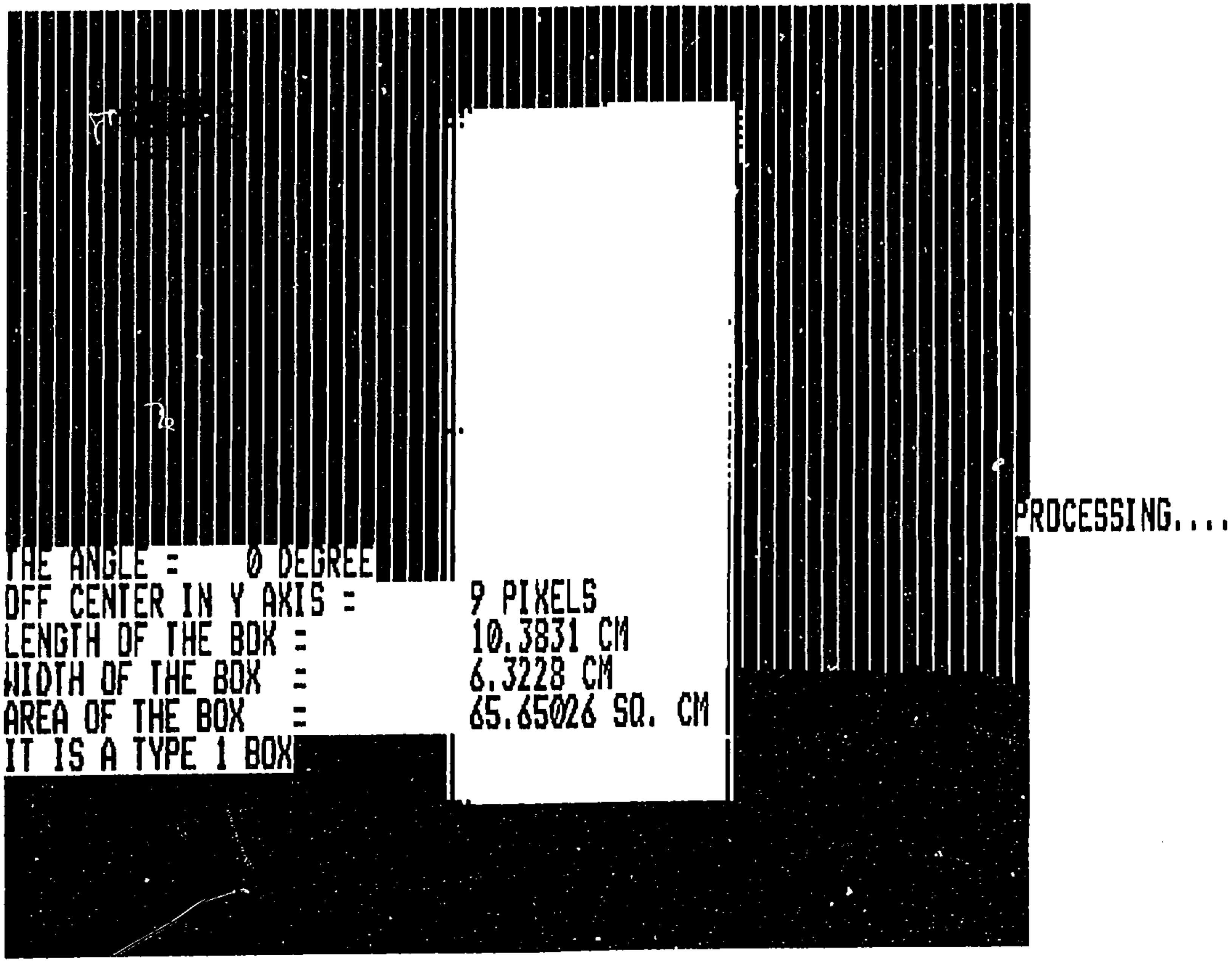
```
PROCESSING....
```

```
STL( 1 ) OF LAYER 2 = 1        STW( 1 ) OF LAYER 2 = 1
FINL( 1 ) OF LAYER 2 = 12      FINW( 1 ) OF LAYER 2 = 10
LP( 1 ) OF LAYER 2 = 12        WP( 1 ) OF LAYER 2 = 10
?
```

```
Load type 3 box onto layer 1 and area 1 of pallet
STL= 10 STW= 6 FINL= 12 FINW= 10
?
```

```
Load type 2 box onto layer 2 and area 1 of pal. at
STL= 1 STW= 1 FINL= 12 FINW= 10
?
```

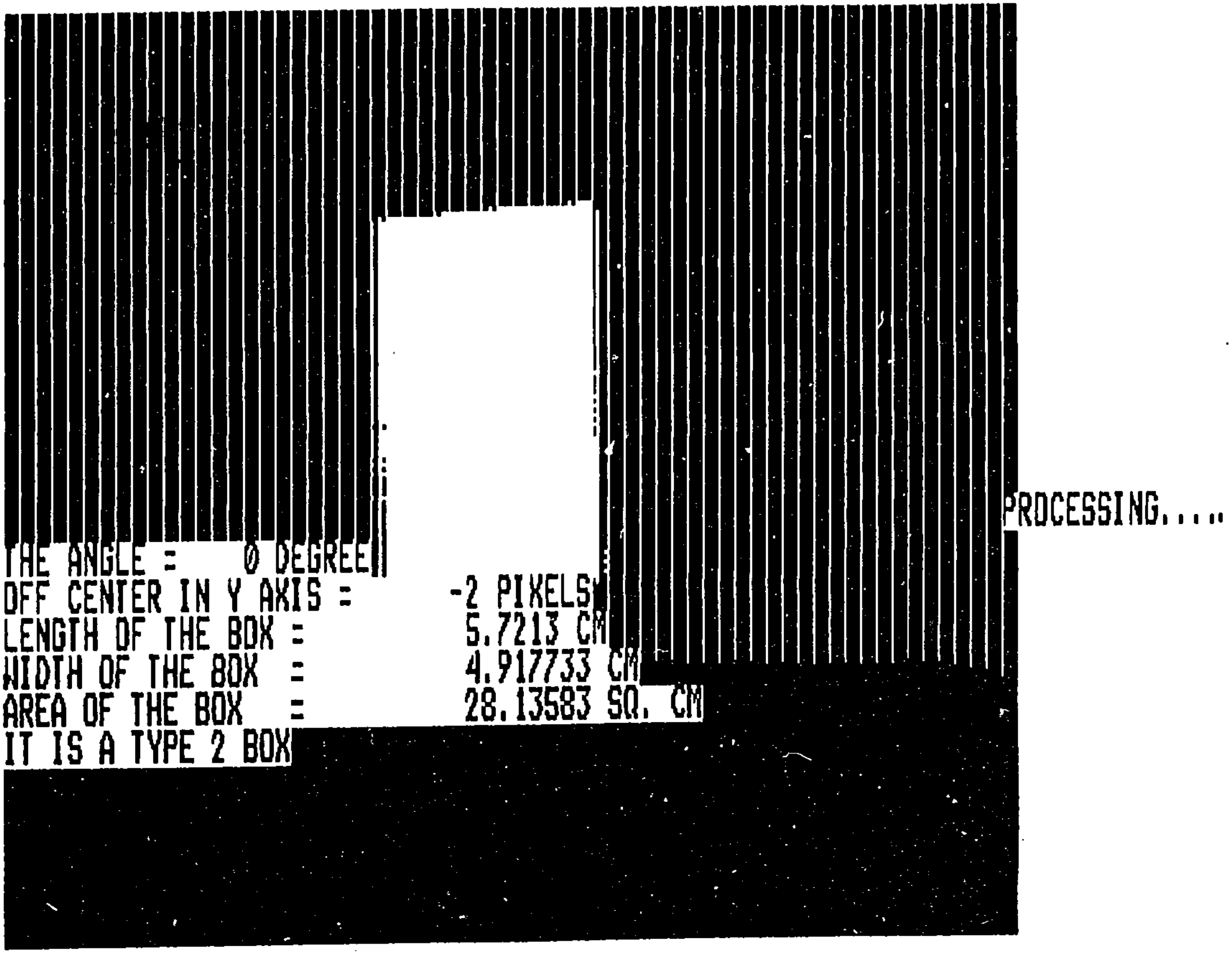
```
STL( 1 ) OF LAYER 2 = 7        STW( 1 ) OF LAYER 2 = 1
FINL( 1 ) OF LAYER 2 = 12      FINW( 1 ) OF LAYER 2 = 10
LP( 1 ) OF LAYER 2 = 6         WP( 1 ) OF LAYER 2 = 10
STL( 2 ) OF LAYER 2 = 1        STW( 2 ) OF LAYER 2 = 6
FINL( 2 ) OF LAYER 2 = 6       FINW( 2 ) OF LAYER 2 = 10
LP( 2 ) OF LAYER 2 = 6         WP( 2 ) OF LAYER 2 = 5
STL( 1 ) OF LAYER 3 = 1        STW( 1 ) OF LAYER 3 = 1
FINL( 1 ) OF LAYER 3 = 6       FINW( 1 ) OF LAYER 3 = 5
LP( 1 ) OF LAYER 3 = 6         WP( 1 ) OF LAYER 3 = 5
?
```



```
THE ANGLE =    0 DEGREE
OFF CENTER IN Y AXIS =       9 PIXELS
LENGTH OF THE BOX =          10.3831 CM
WIDTH OF THE BOX  =          6.3228 CM
AREA OF THE BOX   =          65.65026 SQ. CM
IT IS A TYPE 1 BOX
```

```
PROCESSING....
```

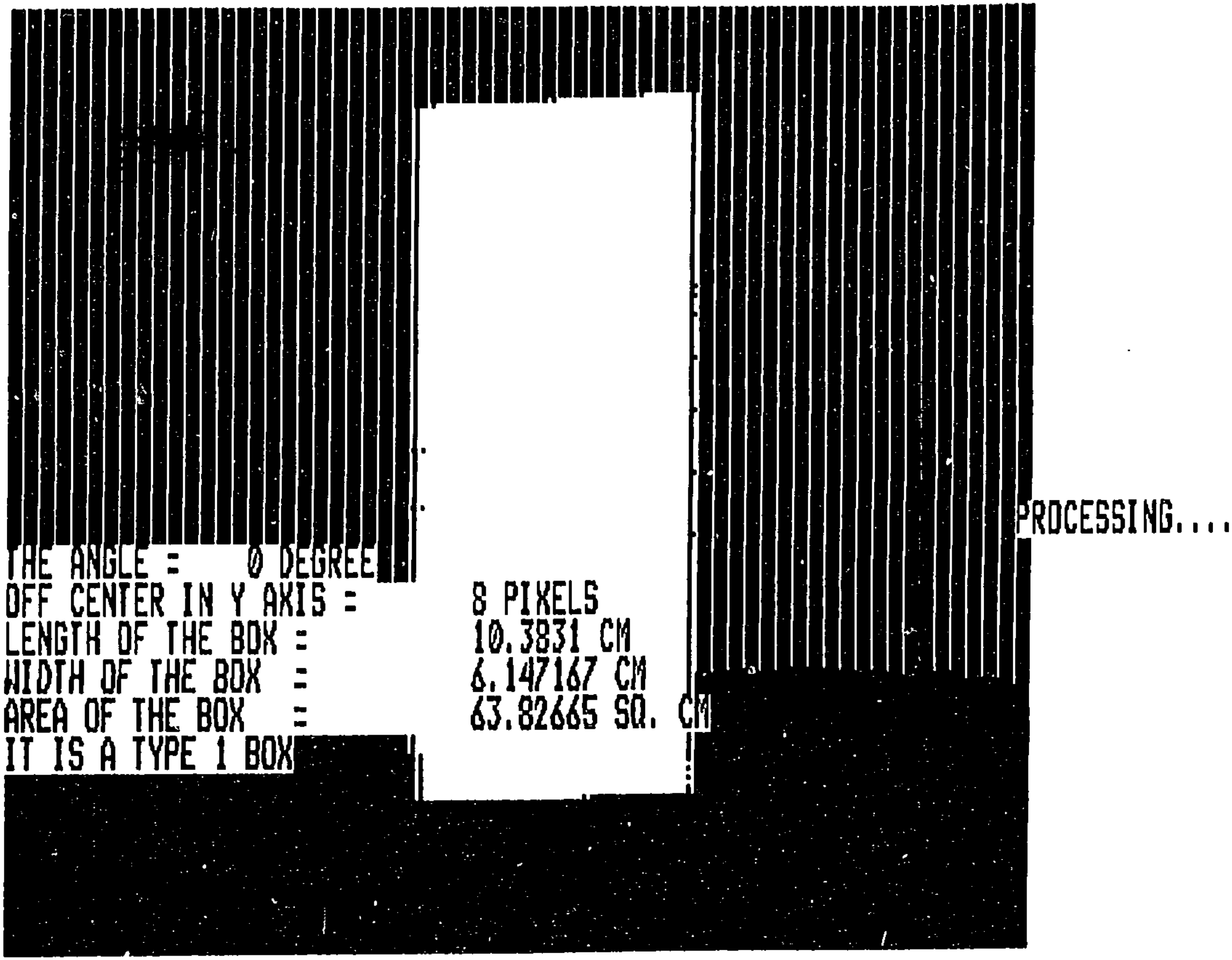
```
NIP  1  - 7
?
Load type 1 box onto layer 2 and area 1 of pallet
STL= 7 STW= 1 FINL= 12 FINW= 10
?
```



```
THE ANGLE =    0 DEGREE
OFF CENTER IN Y AXIS =       -2 PIXELS
LENGTH OF THE BOX =           5.7213 CM
WIDTH OF THE BOX  =           4.917733 CM
AREA OF THE BOX   =           28.13583 SQ. CM
IT IS A TYPE 2 BOX
```

```
PROCESSING.....
```

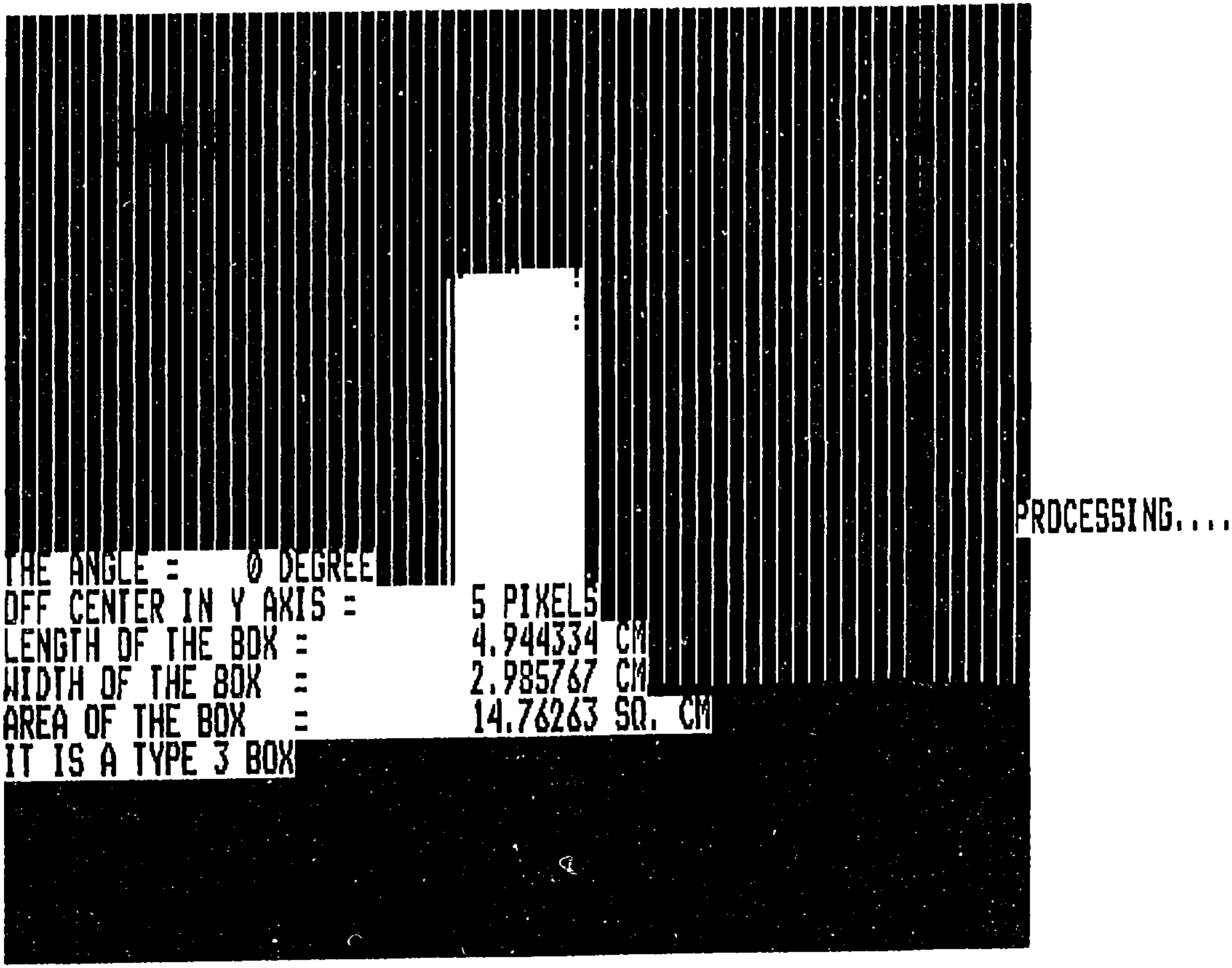
```
STL( 1 ) OF LAYER 2 = [illegible]     STW( 1 ) OF LAYER 2 = [illegible]
FINL( 1 ) OF LAYER 2 = 6    FINW( 1 ) OF LAYER 2 = 1.
LP( 1 ) OF LAYER 2 = 6      WP( 1 ) OF LAYER 2 = 5
STL( 1 ) OF LAYER 3 = 7     STW( 1 ) OF LAYER 3 = 1
FINL( 1 ) OF LAYER 3 = 12    FINW( 1 ) OF LAYER 3 = 10
LP( 1 ) OF LAYER 3 = 6      WP( 1 ) OF LAYER 3 = 10
STL( 2 ) OF LAYER 3 = 1     STW( 2 ) OF LAYER 3 = 1
FINL( 2 ) OF LAYER 3 = 5    FINW( 2 ) OF LAYER 3 = 5
LP( 2 ) OF LAYER 3 = 6      WP( 2 ) OF LAYER 3 = 5
?
Load type 2 box onto layer 2 and area 1 of pallet
STL= 1 STW= 6 FINL= 5 FINW= 10
?
```



```
THE ANGLE =    0 DEGREE
OFF CENTER IN Y AXIS =        8 PIXELS
LENGTH OF THE BOX =           10.3831 CM
WIDTH OF THE BOX  =           6.147167 CM
AREA OF THE BOX   =           63.82665 SQ. CM
IT IS A TYPE 1 BOX
```

```
PROCESSING....
```

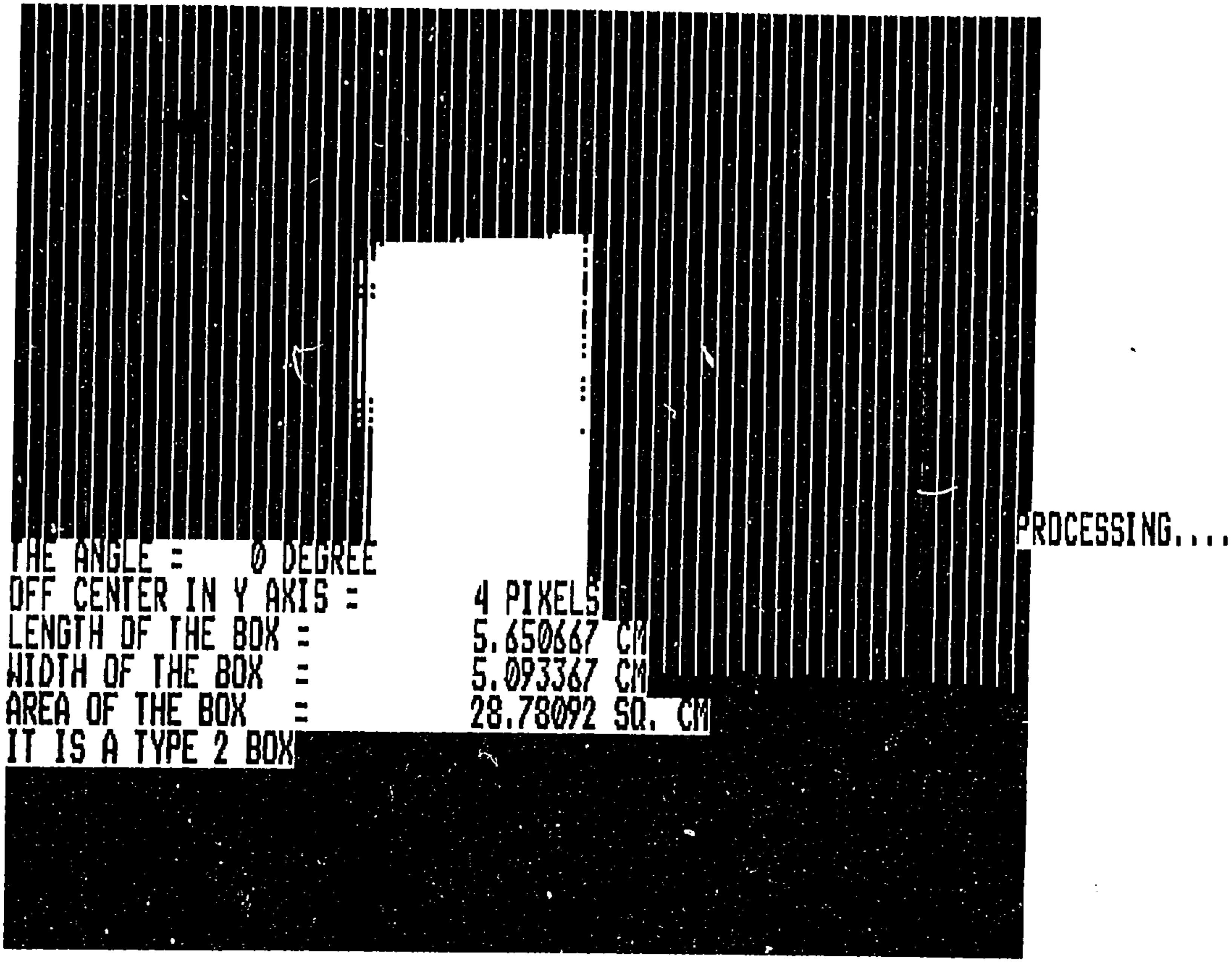
```
STL [illegible] OF LAYER 3 [illegible]     STW [illegible]
FINL( 1 ) OF LAYER 3 = 12     FINW( 1 ) OF [illegible]
LF( 1 ) OF LAYER 3 = 12       WF [illegible] OF LAYER 3 = 10
?
Load type 1 [illegible] layer 3 and area 1 of pallet
STL= [illegible] STW= [illegible] FINL= 12 FINW= 10
```



```
PROCESSING....
THE ANGLE =     0 DEGREE
OFF CENTER IN Y AXIS =        5 PIXELS
LENGTH OF THE BOX =           4.944334 CM
WIDTH OF THE BOX  =           2.985767 CM
AREA OF THE BOX   =           14.76263 SQ. CM
IT IS A TYPE 3 BOX
```

```
STL( 1 ) OF [illegible]
[illegible]
[illegible] S
[illegible] type [illegible]
```

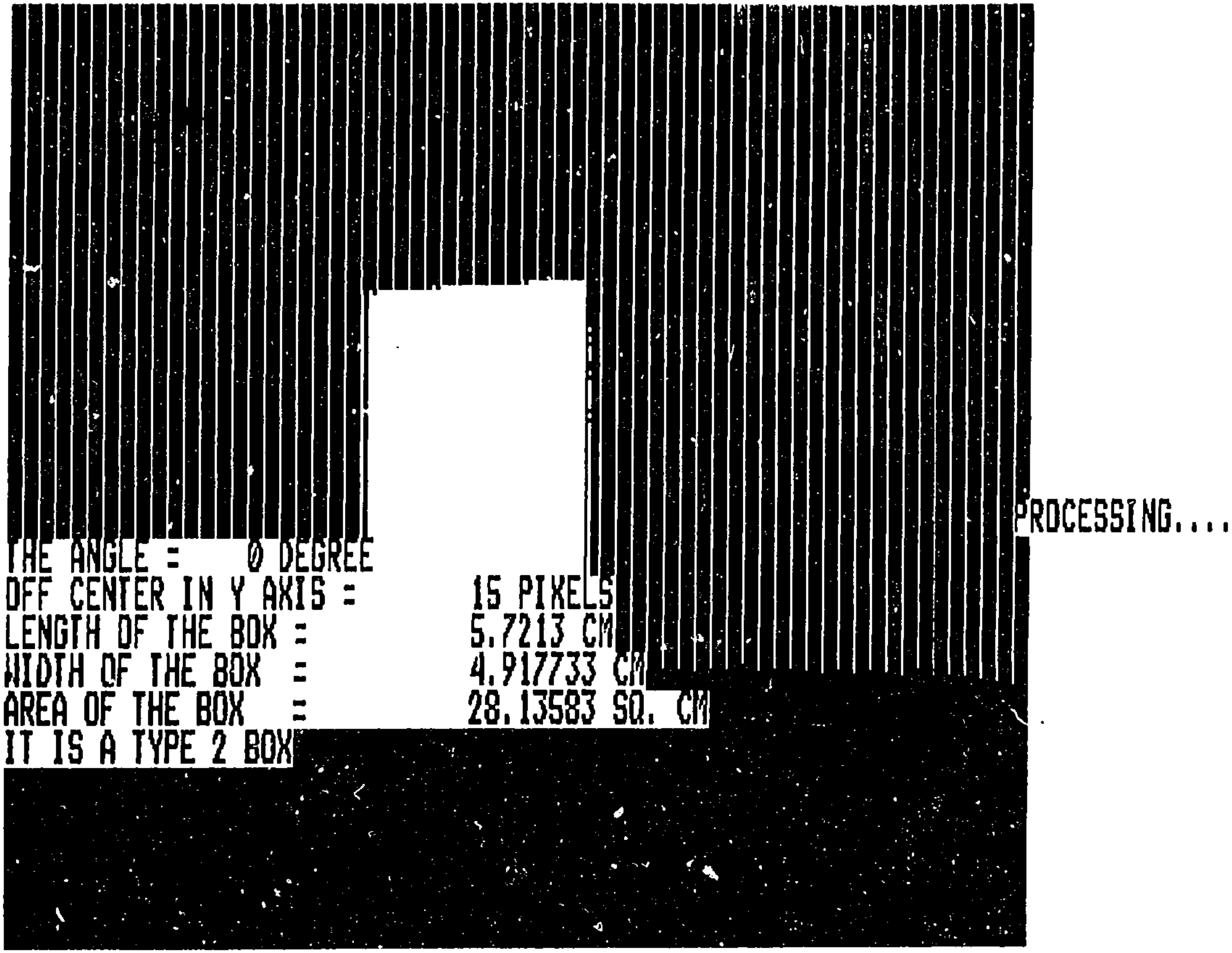
```
                                   STW( 1 ) OF LAYER 3 = [illegible]
            OF LAYER 3 = 12         FINW( 1 ) OF LAYER 3 = 10
          OF LAYER 3 = 6           WP( 1 ) OF LAYER 3 = 5
STL( 2 ) OF LAYER 3 = 10            STW( 2 ) OF LAYER 3 = 1
FINL( 2 ) OF LAYER 3 = 12           FINW( 2 ) OF LAYER 3 = 5
LP( 2 ) OF LAYER 3 = 3             WP( 2 ) OF LAYER 3 = 5
?
```



```
THE ANGLE =      0 DEGREE
OFF CENTER IN Y AXIS =        4 PIXELS
LENGTH OF THE BOX =           5.650667 CM
WIDTH OF THE BOX  =           5.093367 CM
AREA OF THE BOX   =           28.78092 SQ. CM
IT IS A TYPE 2 BOX
```

```
PROCESSING....
```

```
WIP( 1 )= 3
?
Load type 2 box onto layer 3 and area 1 of pallet
STL= 7 STW= 6 FINL= 12 FINW= 10
?
```



```
THE ANGLE =    0 DEGREE
OFF CENTER IN Y AXIS =        15 PIXELS
LENGTH OF THE BOX =           5.7213 CM
WIDTH OF THE BOX  =           4.917733 CM
AREA OF THE BOX   =           28.13583 SQ. CM
IT IS A TYPE 2 BOX
```

```
PROCESSING....
```

```
STL( 1 ) OF LAYER 3 = 10.     STW( 1 ) OF LAYER 3 = 1
FINL( 1 ) OF LAYER 3 = 12     FINW( 1 ) OF LAYER 3 = 5
LP( 1 ) OF LAYER 3 = 3       WP( 1 ) OF LAYER 3 = 5
?
Load type 3 box onto layer 3 and area 1 of pallet
STL= 10 STW= 1 FINL= 12 FINW= 5
?
```

```
?
BOXES ON PALLET ARE AS FOLLOWS :
 1 : 2
 2 : 2
 3 : 2
 4 : 3
 5 : 3
 6 : 2
 7 : 1
 8 : 2
 9 : 1
 10 : 3
 11 : 2
 12 : 3
MAXIMUM WORK IN PROCESS IS AS FOLLOWS :
 1 : 3
THE PALLET IS FULL AND THE UTILIZATION IS  100 PERCENT
THE TOTAL TIME IS00:[illegible]5:54
?
```

VITA AUCTORIS

1965	Born In Hong Kong on the 6th of May.
1984	Completed high school education from Hon. W. C. Kennedy Collegiate Institute, Windsor, Ontario, Canada.
1988	Graduated from University of Windsor, Windsor, Ontario, Canada, with a B. A. Sc. (Honors) Degree in Industrial Engineering.
1990	Currently a candidate for the M. A. Sc Degree in Industrial Engineering at the University of Windsor, Windsor, Ontario, Canada.